IAB Presentation

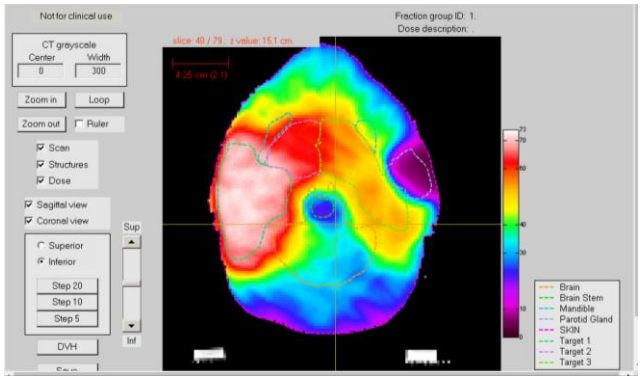
Hyeonjeong Cho
(Department of Biomedical Sciences)

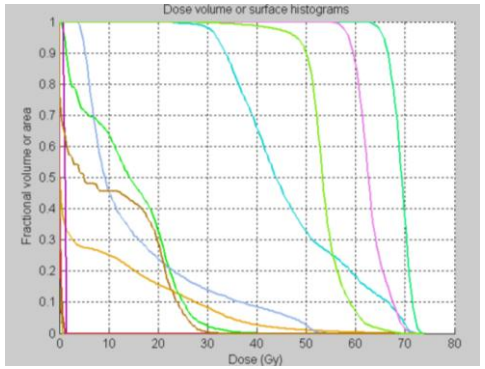
Introduction

- Clinical operation: iterative, back-end-forth in conventional RT

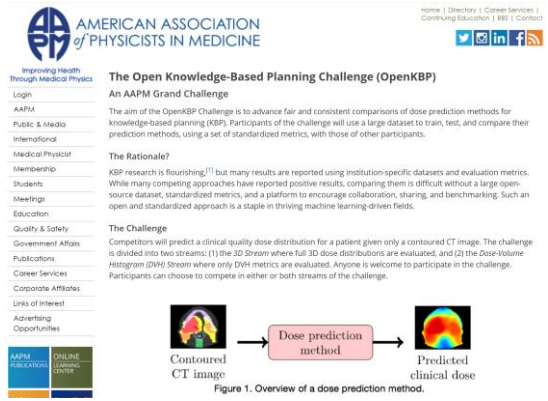
  → Machine-learning, especially **GAN** (Generative Adversarial Network) AI programming algorithm could a solution for these issues which is adjusted manually and based on complex but meaningful rules.

RT Procedure



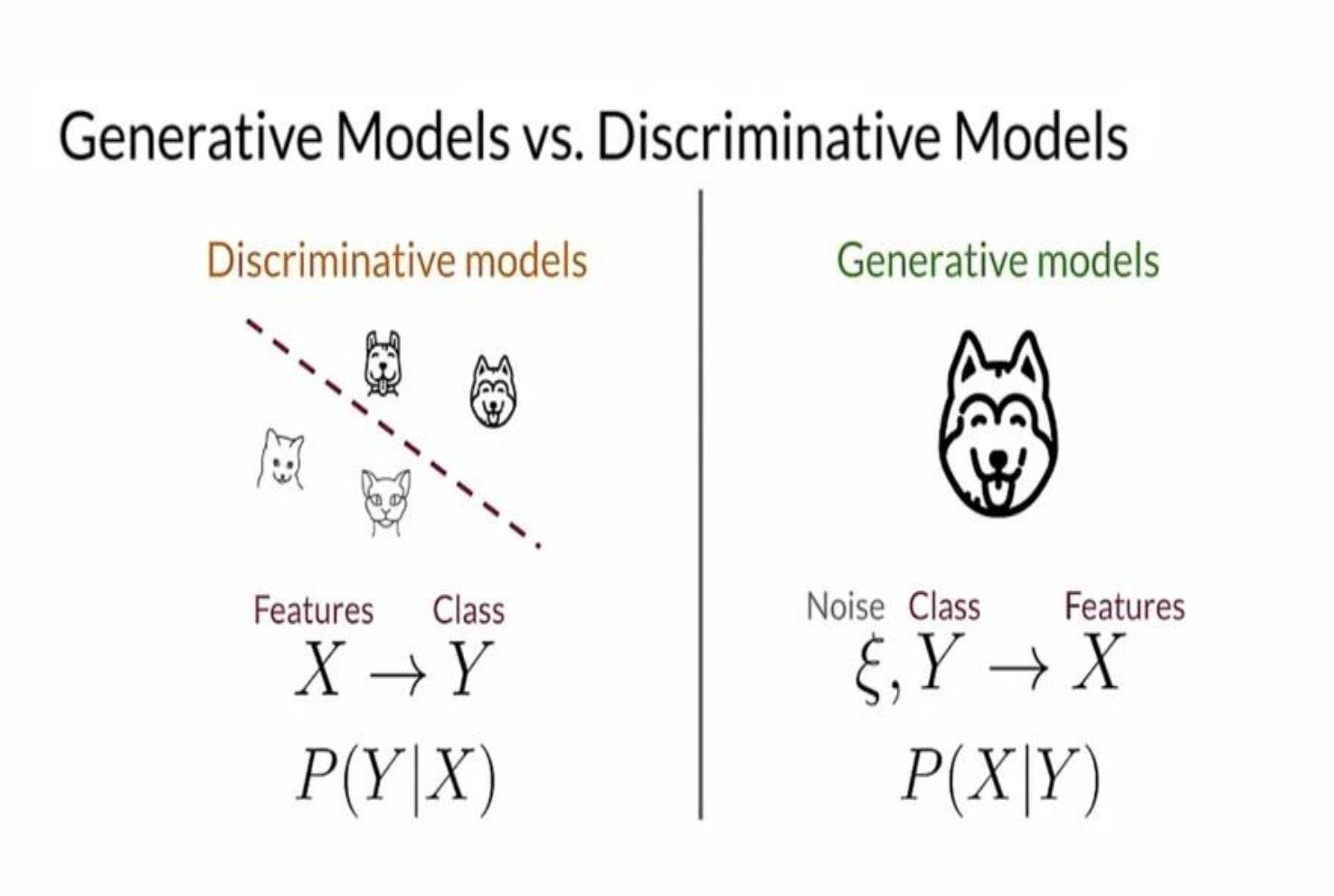


DVH

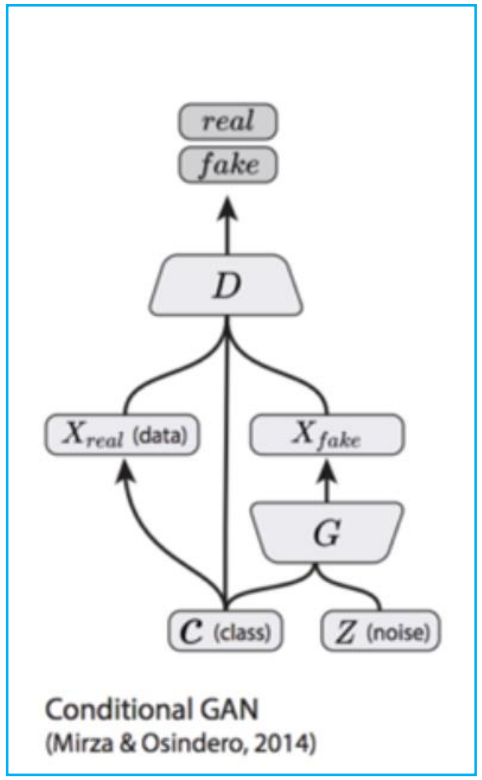


RTDosePrediction in Open-KBP 1st competition

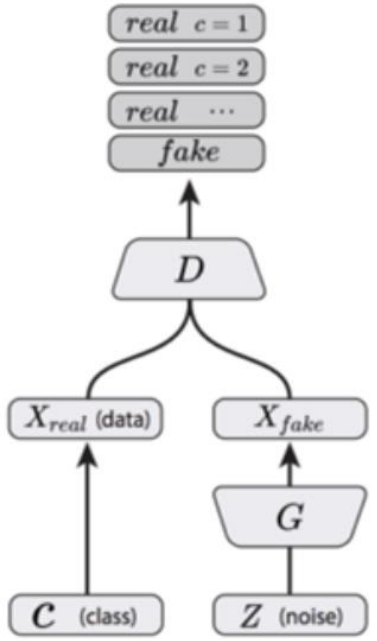
GAN (Generative Adversarial Network)



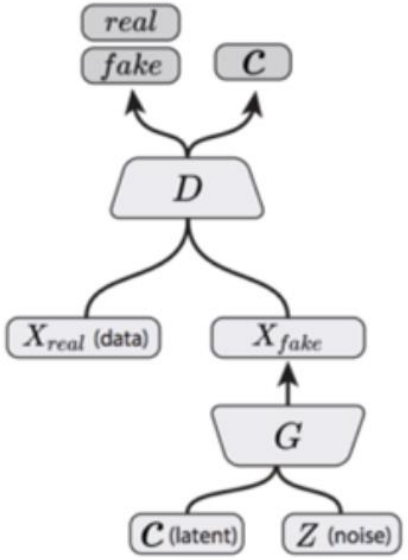
Semi-conditional GANs



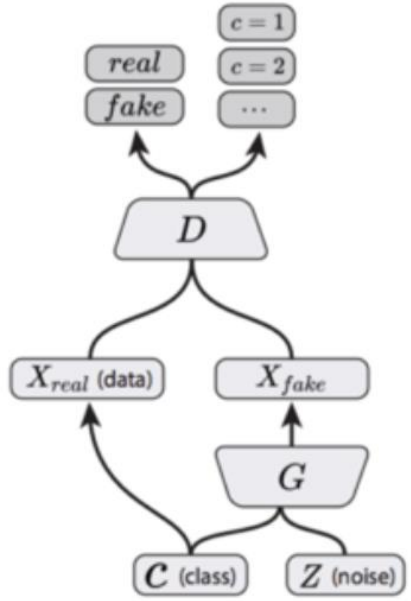
Conditional GAN
(Mirza & Osindero, 2014)



Semi-Supervised GAN
(Odena, 2016; Salimans, et al., 2016)



InfoGAN
(Chen, et al., 2016)



AC-GAN
(Present Work)

Conditional GAN (cGAN)

$$\min_G \max_D \mathcal{L}(D, G)$$

where

$$\mathcal{L}(D, G) = \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}} \left[\log(1 - D(G(\mathbf{z}, \mathbf{c}), \mathbf{c})) \right] + \mathbf{E}_{\mathbf{x} \sim p_{\mathbf{data}}} \left[\log D(\mathbf{x}, \mathbf{c}) \right] + \lambda \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{data}}, \mathbf{z} \sim p_{\mathbf{z}}} \left[\|\mathbf{x} - G(\mathbf{z}, \mathbf{c})\|_1 \right].$$

Mehdi Merza, 2014

- $z \sim P_z$: a sample from a Gaussian input
- $\mathrm{X} \sim P_{data}$: the distribution of *(real)* delivered plans
- $G(z, c)$: a predicted dose distribution *(fake)*

Overview

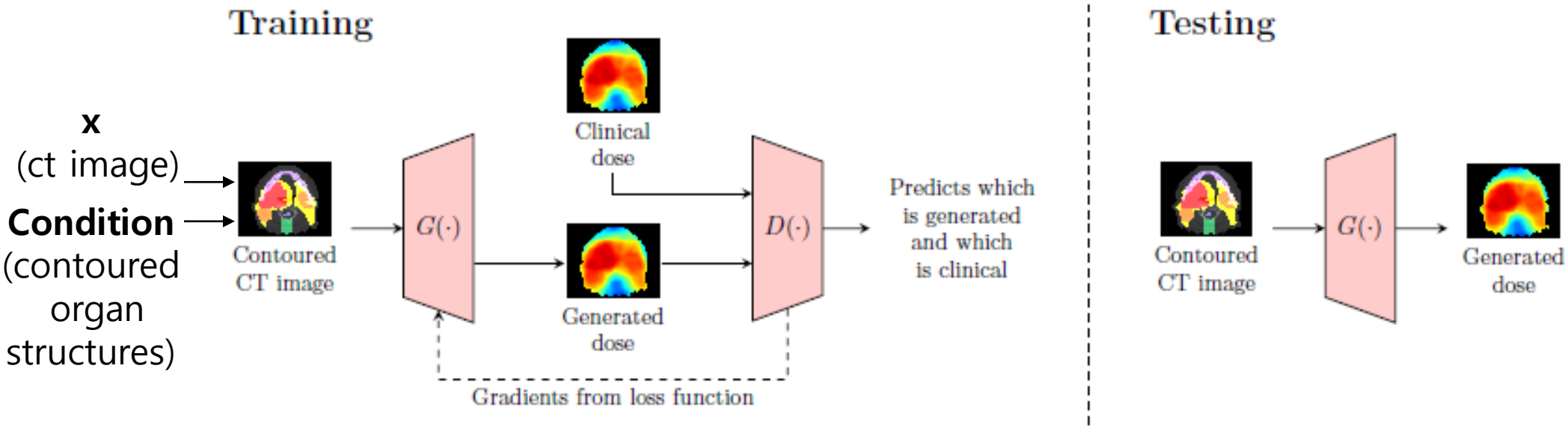


Figure 2: Overview of the GAN training and testing phases.

Data

1. Dose

- Fluence-based treatments plans with similar degrees of fluence complexity (Craft, 2007)
- **1)** a dose deposition matrix using the same parameters in CERR (Computatioanl Environment for Radiotherapy Research) (Deasy, 2003)
- **2)** from nine equispaced coloanar fields at 0, 40, ..., 320 degree with 6 MV step-and-shoot intensity-modulated radiation therapy in 35 fractions.
- Full 3D distributioin, 3D tensor as 128*128*128 voxels, unit of Gy

2. CT

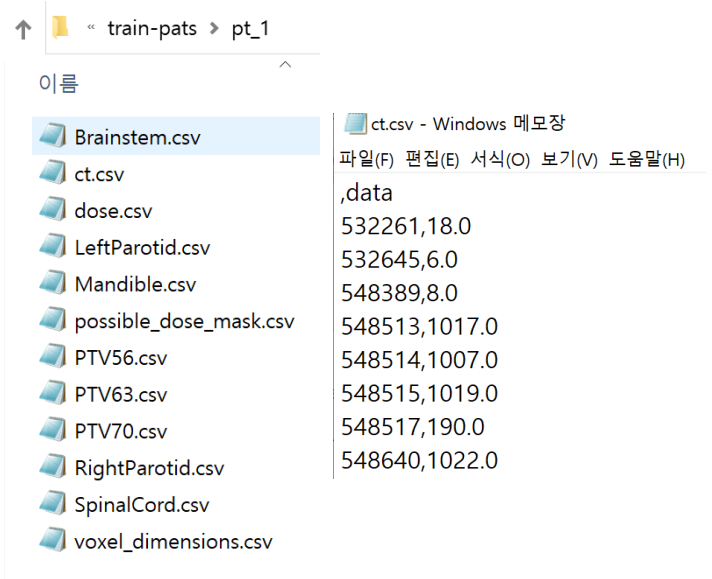
- 3.5 mm x 3.5 mm x 2 mm
- 3D tensor as 128*128*128 voxels

3. Structure

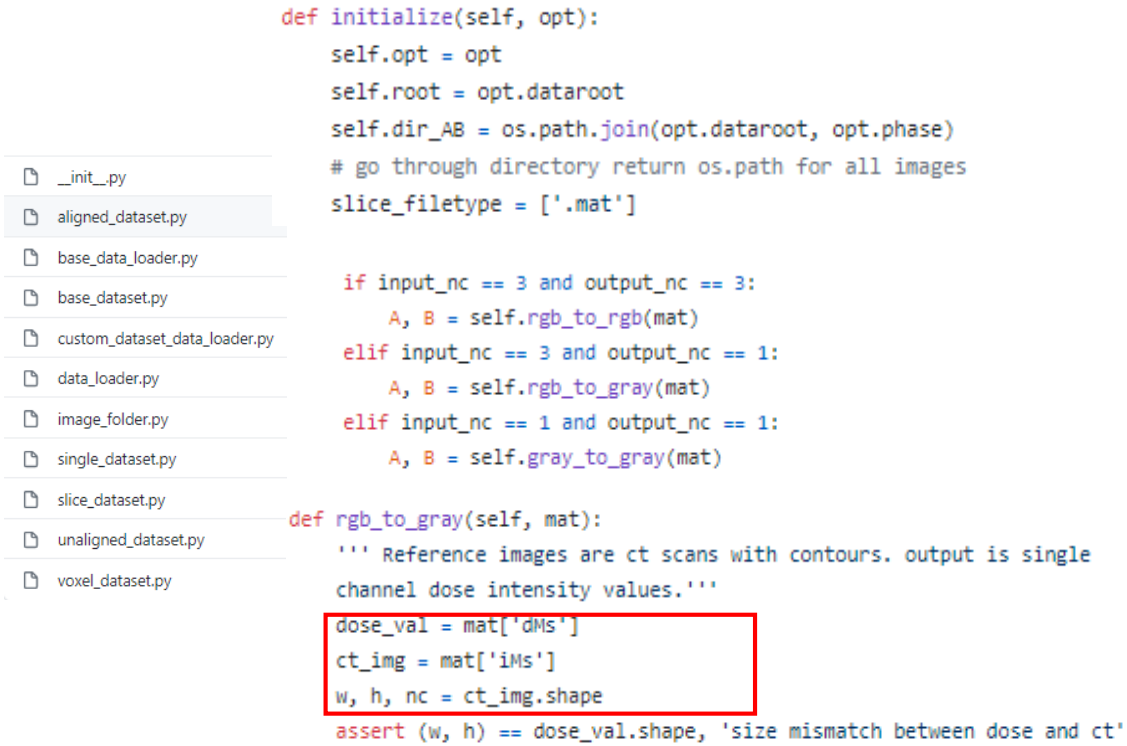
- **OAR (Organs-at-risk):** segmented by the brainstem, spinal cord, right parotid, left parotid, larynx, esophagus, and mandible
- **Target**: the gross disease (PTV70), intermediate-risk target volumes (PTV63), elective target volumes (PTV56)

Data preprocessing

- Mapping single numbers to 3D (i.e., x-y-z) coordinate systems



Provided data



Provided loder

```
for pat = 2:2

    for z=42:43
        dMs = rgb2gray(imread(['D:₩ClassWopen-kbpWpt', num2str(pat), '_dose', num2str(z), '_img.png']));
%         frame = dMs;
        dMs_size = size(dMs);
        dMs_size_w = dMs_size(1);
        dMs_size_h = dMs_size(2);
        dMs = dMs (dMs_size_w/2 - 127:dMs_size_w/2 +128, dMs_size_h/2 - 127:dMs_size_h/2 +128, :);
%         figure, imshow(dMs,[]);

        struc = {'ct','Brainstem', 'LeftParotid', 'RightParotid', 'Mandible', 'SpinalCord', 'PTV70', 'PTV
        ContouredCT = imread(['D:₩ClassWopen-kbpWpt', num2str(pat), '_ct', num2str(z), '_img.png']);
        ContouredCT_size = size(ContouredCT);
        ContouredCT_size_w = ContouredCT_size(1);
        ContouredCT_size_h = ContouredCT_size(2);
        ContouredCT = ContouredCT (ContouredCT_size_w/2 - 127:ContouredCT_size_w/2 +128, ContouredCT_size
%         figure, imshow(ContouredCT,[]);
        for x = 1 : 256
            for y = 1 : 256
                for num_struc = 1 : length(struc)
                    iMs = rgb2gray(imread(['D:₩ClassWopen-kbpWpt', num2str(pat), '_', struc{num_struc}(1:
                    iMs_size = size(iMs);
                    iMs_size_w = iMs_size(1);
                    iMs_size_h = iMs_size(2);
                    iMs = iMs (iMs_size_w/2 - 127:iMs_size_w/2 +128, iMs_size_h/2 - 127:iMs_size_h/2 +128
%                     figure, imshow(iMs,[]);
                    if num_struc == 1
                        ContouredCT (y,x,3) = iMs(y,x);
                    if num_struc == 1
                        ContouredCT (y,x,3) = iMs(y,x);
                    elseif num_struc == 2      % Brainstem
                        if iMs(y,x) ~= 255
                            ContouredCT (y,x,1) = 0;
                            ContouredCT (y,x,2) = 125;
                        end
                    elseif num_struc == 3      % LeftParotid
                        if iMs(y,x) ~= 255
                            ContouredCT (y,x,1) = 0;
                            ContouredCT (y,x,2) = 190;
                        end
                    elseif num_struc == 4      % RightParotid
                        if iMs(y,x) ~= 255
                            ContouredCT (y,x,1) = 0;
                            ContouredCT (y,x,2) = 190;
                        end
                    elseif num_struc == 5      % Mandible
                        if iMs(y,x) ~= 255
                            ContouredCT (y,x,1) = 0;
                            ContouredCT (y,x,2) = 168;
                        end
                    elseif num_struc == 6      % SpinalCord
                        if iMs(y,x) ~= 255
                            ContouredCT (y,x,1) = 0;
                            ContouredCT (y,x,2) = 147;
                        end
                    elseif num_struc == 7      % PTV70
                        if iMs(y,x) ~= 0
                            ContouredCT (y,x,1) = 255;
```

Date Processing

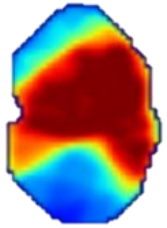
Table 1: Colors assigned to each voxel. Voxels that were classified as both OAR and target were assigned nonzero green and red channel values, respectively.

Structure	Red Channel	Green Channel	Blue Channel
Brainstem	0	125	CT Grayscale
Spinal Cord	0	147	CT Grayscale
Right Parotid	0	190	CT Grayscale
Left Parotid	0	190	CT Grayscale
Larynx	0	233	CT Grayscale
Esophagus	0	212	CT Grayscale
Mandible	0	168	CT Grayscale
limPostNeck	0	255	CT Grayscale
PTV70	255	0	CT Grayscale
PTV63	205	0	CT Grayscale
PTV56	155	0	CT Grayscale
Unclassified	0	0	CT Grayscale
Empty Space	0	0	0

ct

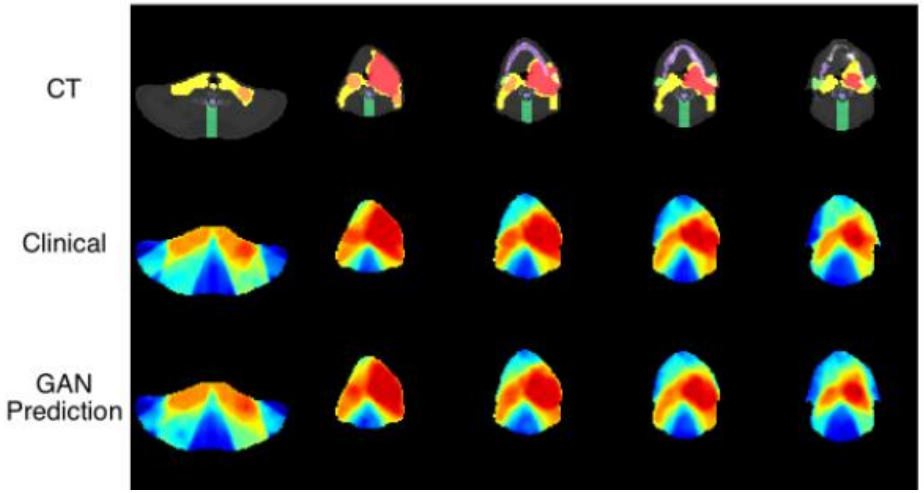
structure

Contoured ct



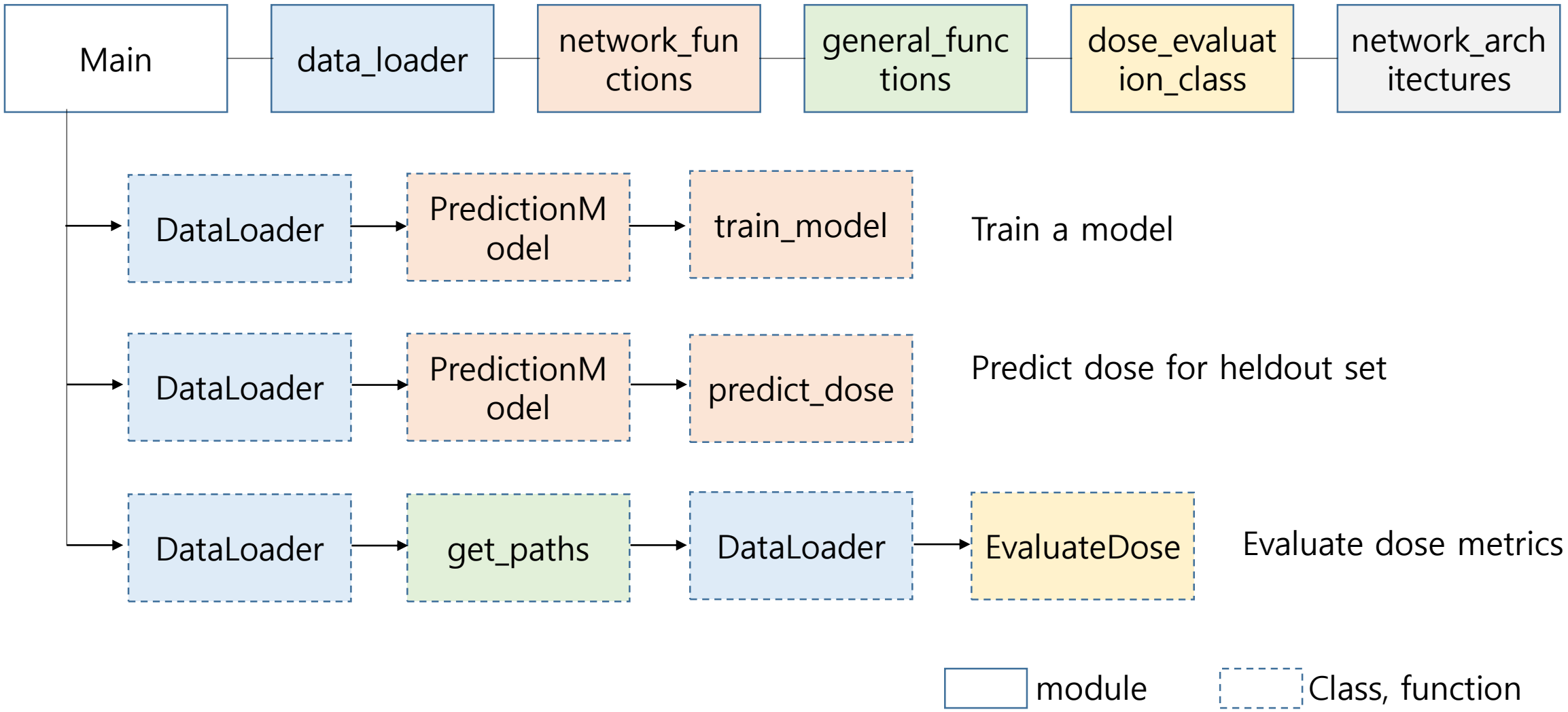
dose

Reference



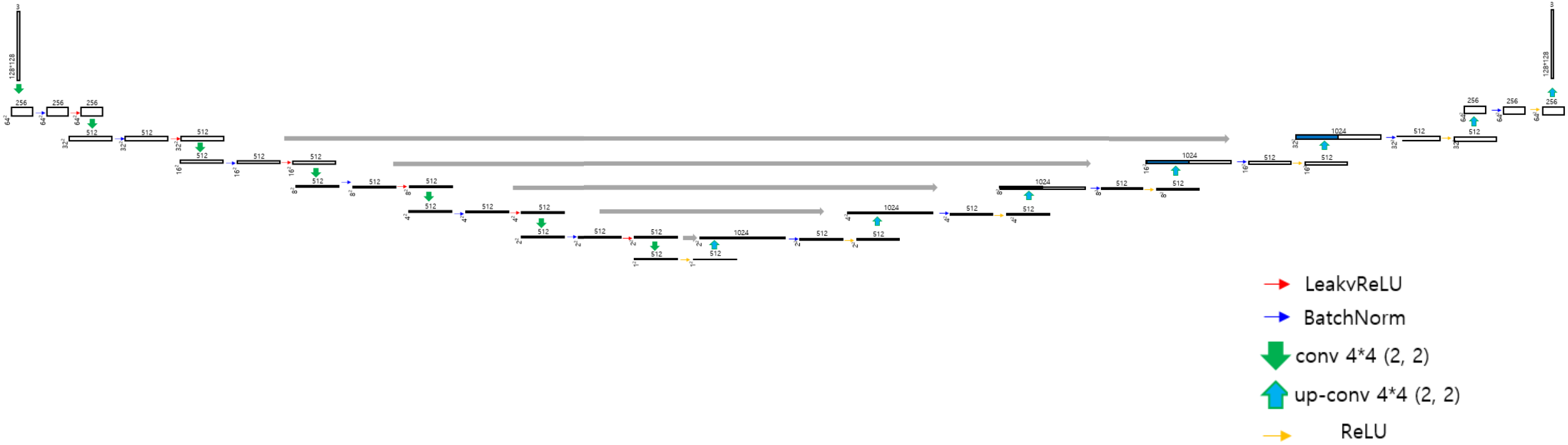
Knowledge-based automated planning with 3D generative adversarial networks, Aaron Babier, 2019

Code Tree


Main
data_loader
network_functions
general_functions
dose_evaluation_class
network_architectures
DataLoader
PredictionModel
train_model
Train a model
DataLoader
PredictionModel
predict_dose
Predict dose for heldout set
DataLoader
get_paths
DataLoader
EvaluateDose
Evaluate dose metrics
module
Class, function

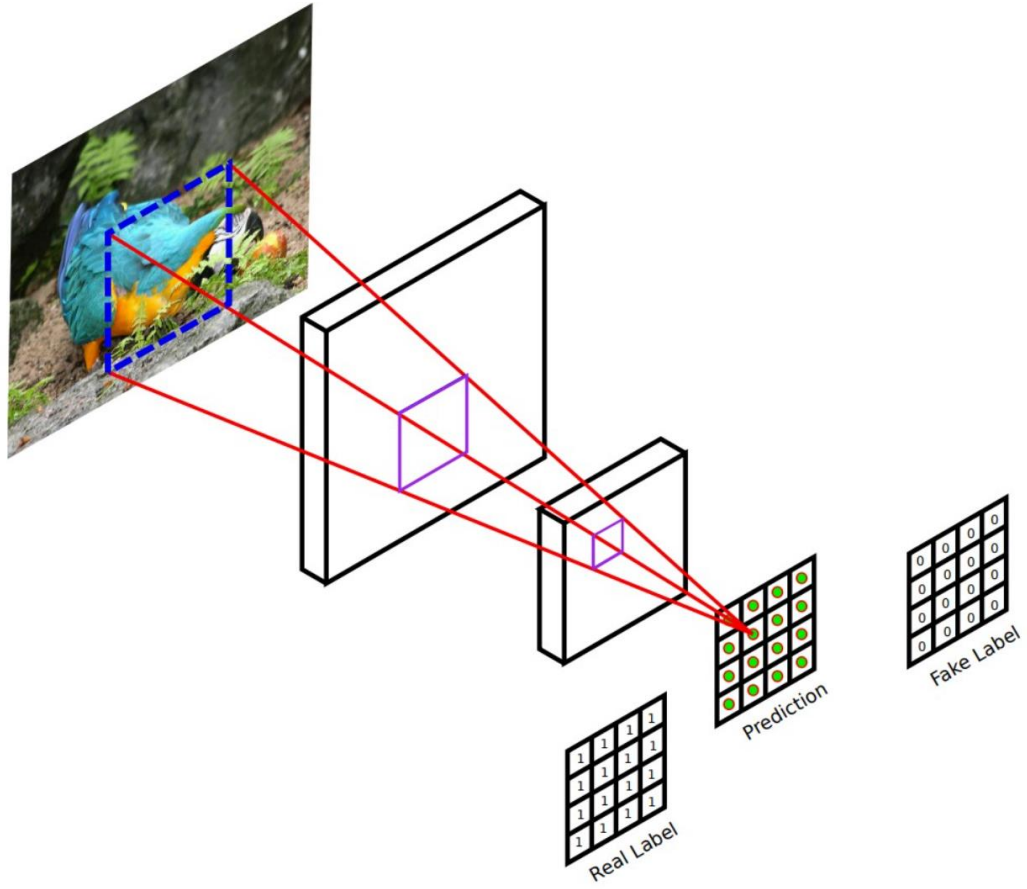
Model Framework

- Generator



Model Framework

- Discriminator



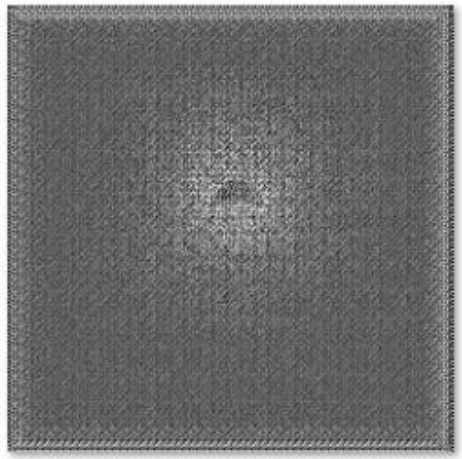

```
conv = nn.Conv3d
norm_layer = functools.partial(nn.BatchNorm3d, affine=True)
```

```
layer1 = conv(4, 64, kernel_size=4, stride=2, padding=1)
layer2 = nn.LeakyReLU(0.2, True)
layer3 = conv( 64 * 1, 64 * 2, kernel_size=4, stride=2, padding=1, bias=False)
layer4 = norm_layer(64 * 2)
layer5 = nn.LeakyReLU(0.2, True)
layer6 = conv( 64 * 2, 64 * 4, kernel_size=4, stride=2, padding=1, bias=False)
layer7 = norm_layer(64 * 4)
layer8 = nn.LeakyReLU(0.2, True)
# Final Conv2d: 2 ** n * ndf -> 1, 4x4 kernels
layer9 = conv(64 * 4, 1, kernel_size=4, stride=1, padding=1)
sig= nn.Sigmoid()
```

Training: 1 epoch

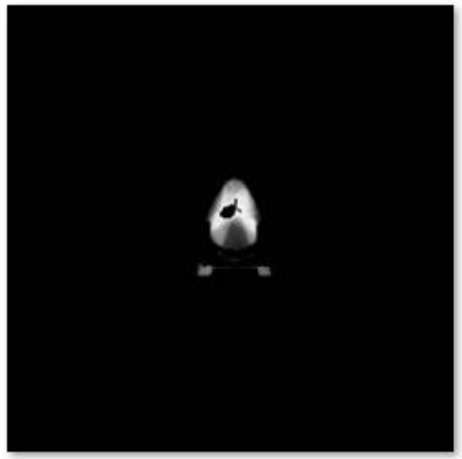
- Generator loss: 0.6383, Discriminator loss: 0.6435 *(real)*, 0.7777 *(fake)*

```
IPdb [16]: print(errors)
OrderedDict([('G_GAN', 0.6382938623428345), ('G_L1', 50421.19140625), ('D_real',
0.6434733271598816), ('D_fake', 0.7771273851394653)])
```



epoch001_fake_B.png

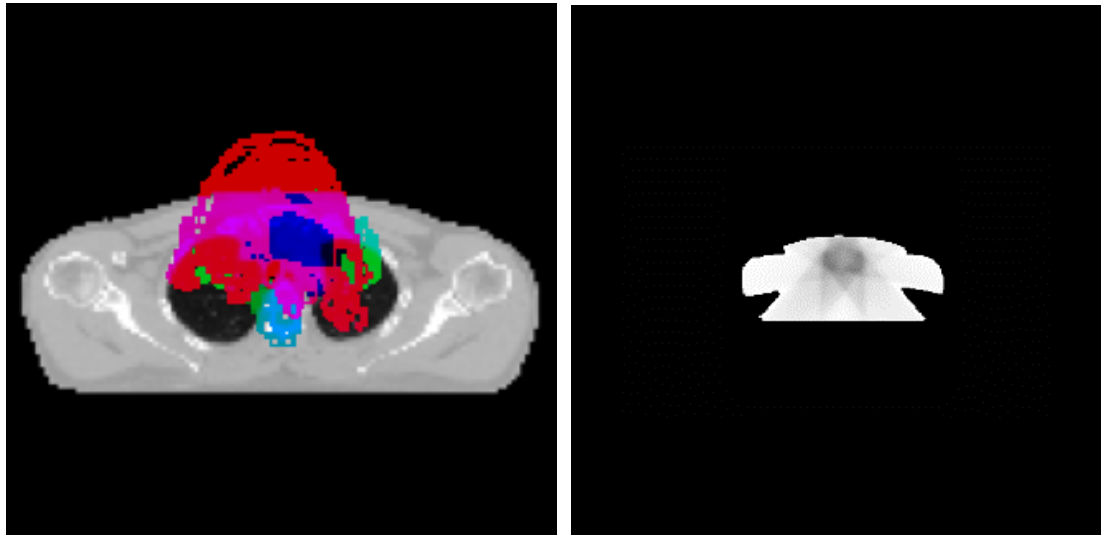
epoch001_real_A.png

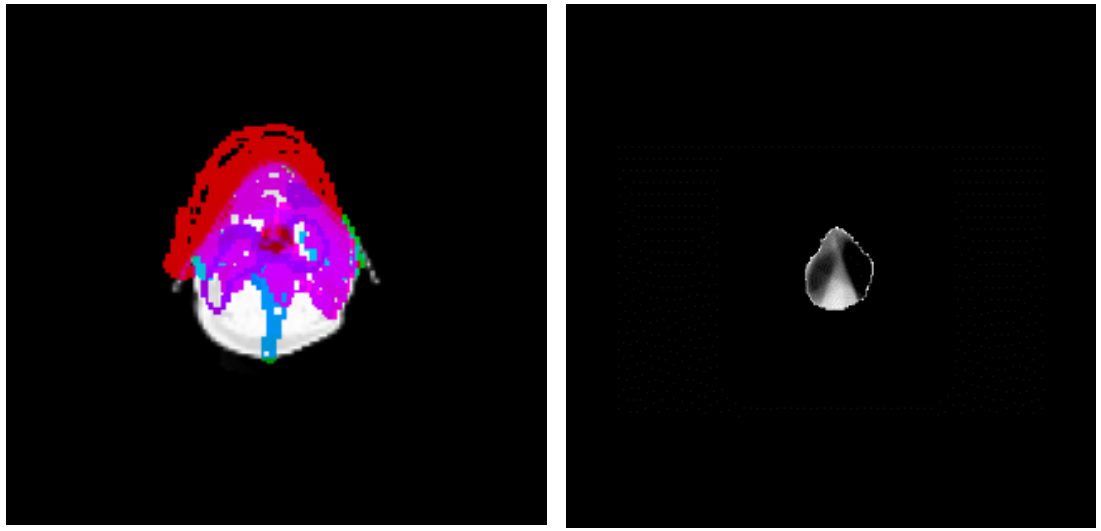


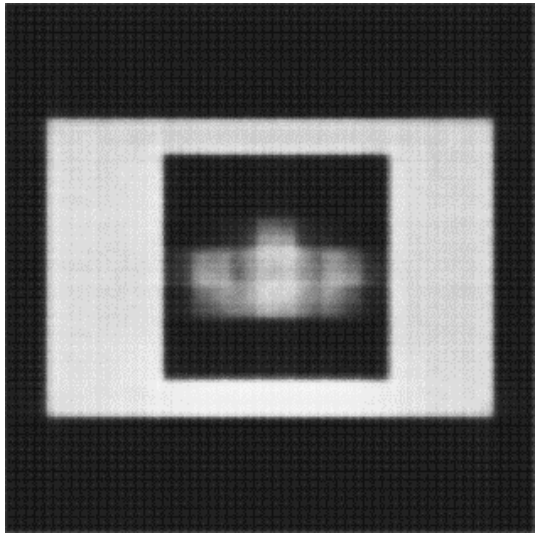
epoch001_real_B.png

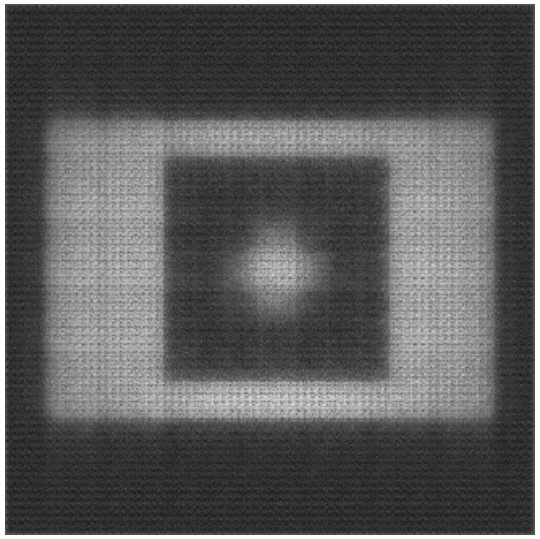
- **GPU** accelation needed for more training

Generated data









Hyper-parameters

Learning rate: 0.002
Lambda: 100
Batch-size: 64
Epoch: 2000
Cost function: BCE
Optimization: Adam

Evaluation

- Dose criteria: D_{max}, D_{99}, $D_{0.1cc}$ ..

Structure	Criteria
Brainstem	$\mathcal{D}_{max} \leq 54$ Gy
Spinal Cord	$\mathcal{D}_{max} \leq 48$ Gy
Right Parotid	$\mathcal{D}_{mean} \leq 26$ Gy
Left Parotid	$\mathcal{D}_{mean} \leq 26$ Gy
Larynx	$\mathcal{D}_{mean} \leq 45$ Gy
Esophagus	$\mathcal{D}_{mean} \leq 45$ Gy
Mandible	$\mathcal{D}_{max} \leq 73.5$ Gy
PTV70	$\mathcal{D}_{99} \geq 66.5$ Gy
PTV63	$\mathcal{D}_{99} \geq 59.9$ Gy
PTV56	$\mathcal{D}_{99} \geq 53.2$ Gy

Table 2: The planning criteria used for evaluation

Table 3: For each KBAP approach, the percentage of clinical criteria that passed and failed compared to the corresponding clinical plans.

			Unscaled						Scaled					
			2D-RGB		2D-dose		3D-dose		2D-RGB′		2D-dose′		3D-dose′	
			Pass	Fail	Pass	Fail	Pass	Fail	Pass	Fail	Pass	Fail	Pass	Fail
Clinical	OARs	Pass	63.4	3.2	64.9	1.7	65.8	0.8	60.5	6.1	63.1	3.5	63.4	3.2
		Fail	6.2	27.2	7.9	25.5	7.9	25.5	4.7	28.7	5.9	27.5	4.6	28.8
	Targets	Pass	45.5	23.2	46.9	21.9	43.8	25.0	60.3	8.5	67.9	0.9	68.3	0.4
		Fail	9.8	21.4	9.4	21.9	8.5	22.8	21.4	9.8	30.4	0.9	31.2	0.0
	All ROIs	Pass	58.5	8.7	60.0	7.2	59.7	7.5	60.5	6.7	64.4	2.8	64.7	2.4
		Fail	7.2	25.6	8.3	24.5	8.1	24.7	9.3	23.5	12.6	20.2	11.9	20.9

- Code

```
if mode == 'target':
    # D1
    output['D1'] = np.percentile(_roi_dose, 99)
    # D95
    output['D95'] = np.percentile(_roi_dose, 5)
    # D99
    output['D99'] = np.percentile(_roi_dose, 1)

elif mode == 'OAR':
    # D_0.1_cc
    _roi_size = len(_roi_dose)
    _voxel_size = np.prod(spacing)
    voxels_in_tenth_of_cc = np.maximum(1, np.round(100 / _voxel_size))
    fractional_volume_to_evaluate = 100 - voxels_in_tenth_of_cc / _roi_size * 100
    output['D_0.1_cc'] = np.percentile(_roi_dose, fractional_volume_to_evaluate)
```

Reference

[1] RTDosePrediction in Open-KBP competition
https://www.aapm.org/GrandChallenge/OpenKBP/

[2] 카일라쉬 아히르와. (2019), 실전! GAN 프로젝트 (박진수, 옮김), 위키북스.

[3] Aaron Babier. (2020). OpenKBP: The open-access knowledge-based planning grand challenge, arXiv:2011.14076v1.

[4] R. Mahmood, A. (2018) Automated treatment planning in radiation therapy using generative adversarial networks. Proceedings of Machine Learning Research, 85:1-15.