SPect for Online boron dose verification in bnCt

BA Team (WP3)

19-06-2024

DIPARTIMENTO INTERATENEO DI FISICA

Ongoing work

- More detailed simulation with Geant4
  - Implement the new collimator version
  - Generate some data for Deep Learning models data base (new approaches)
- New approach for tomography reconstruction
  - DeepRecon (hybrid model) → less data required to train the model
  - SPECTnet

Why use DL methods?
Classics likelihood maximization Iterative methods issues:
→ It needs time to reconstruct: at least 10 min with GPU-Nvidia
→ Artificial noise introduction by iteration

DeepRecon (approach 1)

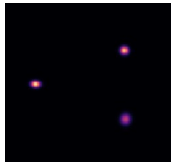
- This reconstruction technique requires an instance of a user-defined PRIOR NETWORK that implements two functions:
- a FIT method that takes in an OBJECT which the network is subsequently fit to
- a PREDICT function that returns the current network prediction

Prior object:

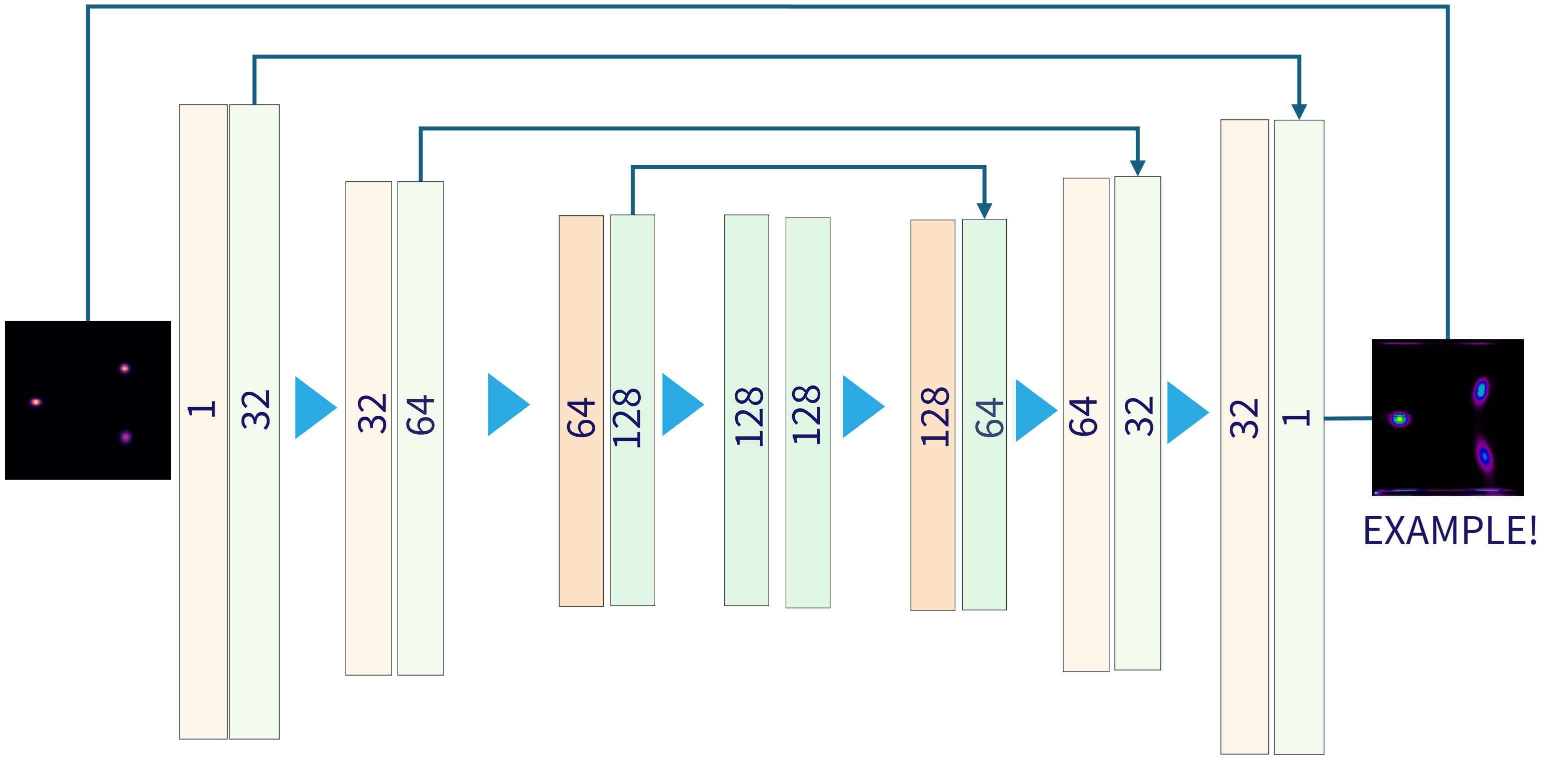
1. It is a reconstructed object (with OSEM or other methods)
2. It is input of the FIT method of the class PRIOR
3. FIT update weights
4. PRIOR predicts updated object with PREDICTION method
5. OUTPUT is the SPECT image

How to

1. Train neural network on the initial reconstructed SPECT image?
2. Update weights?

PRIOR.FIT()

DeepRecon (approach 1)



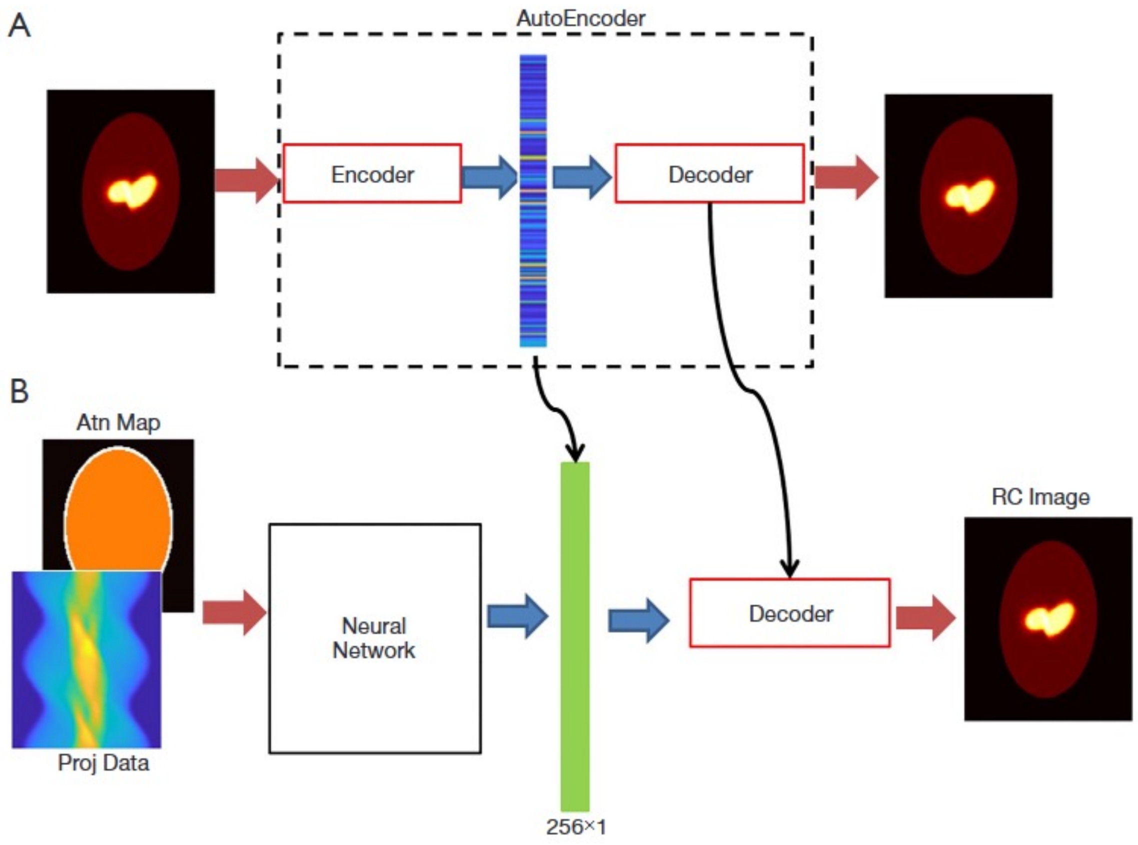

```python
class UNetnew(nn.Module):
    def __init__(self, n_channels=[4, 8, 16, 32, 64]):
        super().__init__()
        self.downward_block1 = downward_block(1, n_channels[0])
        self.downward_block2 = downward_block(n_channels[0], n_channels[1])
        self.downward_block3 = downward_block(n_channels[1], n_channels[2])
        self.downward_block4 = downward_block(n_channels[2], n_channels[3])
        self.downsample_block1 = downsample_block(n_channels[0])
        self.downsample_block2 = downsample_block(n_channels[1])
        self.downsample_block3 = downsample_block(n_channels[2])
        self.downsample_block4 = downsample_block(n_channels[3])
        self.bottleneck_block = bottleneck_block(n_channels[3], n_channels[4])
        self.upsample_block1 = bilinear_upsample_block(n_channels[4], n_channels[3])
        self.upsample_block2 = bilinear_upsample_block(n_channels[3], n_channels[2])
        self.upsample_block3 = bilinear_upsample_block(n_channels[2], n_channels[1])
        self.upsample_block4 = bilinear_upsample_block(n_channels[1], n_channels[0])
        self.upward_block1 = upward_block(n_channels[3])
        self.upward_block2 = upward_block(n_channels[2])
        self.upward_block3 = upward_block(n_channels[1])
        self.final_block = final_block(n_channels[0])

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        x1 = self.downward_block1(x)
        x = self.downsample_block1(x1)
        x2 = self.downward_block2(x)
        x = self.downsample_block2(x2)
        x3 = self.downward_block3(x)
        x = self.downsample_block3(x3)
        x4 = self.downward_block4(x)
        x = self.downsample_block4(x4)
        x = self.bottleneck_block(x)
        x = self.upsample_block1(x) + x4
        x = self.upward_block1(x)
        x = self.upsample_block2(x) + x3
        x = self.upward_block2(x)
        x = self.upsample_block3(x) + x2
        x = self.upward_block3(x)
        x = self.upsample_block4(x) + x1
        x = self.final_block(x)
        return x
```

SPECTnet tomography (approach 2)

SPECTnet autoencoder type network*:

- Network uses as input the sinogram from projections

* Shao W, Rowe SP, Du Y. SPECTnet: a deep learning neural network for SPECT image reconstruction. Ann Transl Med 2021;9(9):819. doi: 10.21037/atm-20-3345



Project Organization – Participants and Costs (INFN Bari 2025)

***Project submitted CNS5:** Compton Imaging System for Advanced Radiotherapy in-vivo monitoring (CISAR)

FTE distribution

G.Pugliese (**RL**, PA)	20 %	(10%)*
G.Iaselli (PO)	50%	
G. Bruno (PO)	10%	
N. Ferrara		(30%)*
D. Ramos		(30%)*
Uhmesh	40%	(30%)*
TOTAL	1.2 FTE	

Richieste

Missioni: 5.0 k

(Lena 2*1kEuro + Milano 4*0.5 kEuro + 1 conferenza 1kEuro)

Finalizzazione meccanica per set-up tomografico: 5 k

Inventario: 2 k (HPC)

Thanks

BA Team @ Collaboration meeting

Dayron.Ramos@ba.infn.it

Reconstruction by DEEP Reconstruction

PIPE-LINE:

1. Pretrain Network with OSEM reconstructed image
2. Use DeepRecon Algorithm to reconstruct: needs the Likelihood of projections and the PRIOR network

```
mu = 0
norm_BP = self.likelihood.system_matrix.compute_normalization_factor()
x = self.prior_network.predict()
x_network = x.clone()
for _ in range(n_iters):
    for j in range(subit1):
        for k in range(n_subsets_osem):
            self.EM_algorithm.object_prediction = nn.ReLU()(x.clone())
            x_EM = self.EM_algorithm(n_iters = 1, n_subsets = n_subsets_osem, n_subset_specific=k)
            x = 0.5 * (x_network - mu - norm_BP / self.rho) + 0.5 * torch.sqrt((x_network - mu - norm_BP / self.rho)**2 + 4 * x_EM * norm_BP / self.rho)
    self.prior_network.fit(x + mu)
    x_network = self.prior_network.predict()
    mu += x - x_network
    self.object_prediction = nn.ReLU()(x_network)
    # evaluate callback
    if self.callback is not None:
        self._compute_callback(n_iter = _, n_subset=None)
return self.object_prediction
```