* MR-CT pairs from RIRE database
  + Training (11 patients, 65%)
    1. Patient\_003
    2. Patient\_004
    3. Patient\_101
    4. Patient\_102
    5. Patient\_103
    6. Patient\_104
    7. Patient\_105
    8. Patient\_106
    9. Patient\_107
    10. Patient\_108
    11. Patient\_109
  + Testing (6 patients, 35%, including PET)
    1. Patient\_001
    2. Patient\_002
    3. Patient\_005
    4. Patient\_006
    5. Patient\_007
    6. Training\_001
  + Don’t have CT

1. Patient\_008 (T1 y PET only)
2. Patient\_009 (T1 y PET only)

Notas importantes:

1. Model name: highres3dnet\_large. Incluir en L40 de highres3dnet\_large.py

|  |  |
| --- | --- |
|  | hires3dnet uses batch norm and no bias term at the final layer by default (<https://github.com/NifTK/NiftyNet/blob/v0.2.2/niftynet/network/highres3dnet_large.py#L140>). You could either a) preprocess the CT to have zero mean and unit variance or b) remove the batch norm and add a bias term to the final layer by fc\_layer = ConvolutionalLayer(..., with\_bias=True, with\_bn=False, ...). para que inference salga entre -1000:1000 |

1. To execute NiftyNet within python

import os  
import sys  
import niftynet  
  
sys.argv=['','inference','-a','net\_segment','-c',os.path.join('D:/I3M/Proyectos/NiftyNet\_skull\_segmentation/model\_vnet','vnet\_config.ini')]  
  
niftynet.main()

1. En el fichero rand\_elastic\_deform.py incluir “params\_numpy[0:int(len(params) / 3)] = 0” en línea 88 para evitar que deforme z dimension. Si se deforma z dimension sale mal.

Sampler

* + - * Hay que hacer padding antes, sobre todo en el eje z, de lo contrario el sampler no agarra los bordes (primer y ultimo slice).
      * Usar 100 samples (o 50) per image para cubrir todo el volumen.
      * 
      * 
      * 

Preprocessing:

1. Padding
2. Normalization
3. Whitenning

Data augmentation:

1. Rotation
2. Scaling
3. Deformation
4. Bias field

**Co-registro:**

En Elastix using FASDGD:

* + - * Registrar CT (moving) y MRI (fixed)
      * Segmentar CT coregistrada (carpeta Labels\_MRI\_space), estos serían las imágenes \_labels.

**Weighted maps:**

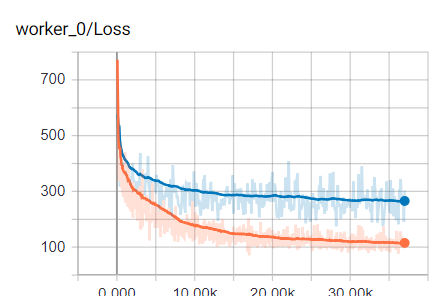
How do your frequency/sampling maps look like? My go to strategy is to create frequency maps by estimating the volume of each class, and then set each pixel of an image to 1/Vol of the pixel’s class. Finally, Gaussian blur the result with a kernel that is half of the patch size. It might be that you are not sampling enough from backgrounds areas, which is hurting you at inference.

**Regression (using weighted maps and not the isampler error\_maps)**

**Highres3dnet with weighted sampling**

1. (Script1) Train 1000 iterations to test different learning rates with regularization = 0
   1. Tried different combinations with batch\_size [1,2,3,4,5,**6**] and learning rate [0.1, **0.01**, 0.001, 0.0001].
      1. Best batch\_size = 6 (improves performance faster)
      2. Best learning\_rate = 0.01
2. (Script2) Try different spatial window\_sizes, samples per volume and batch.
   1. (40,40,40) – batch 6 –samples 1 (Best)
   2. **(40,40,40) – batch 6 –samples 6 (OK)**
   3. (40,40,40) – batch 6 –samples 12
   4. (40,40,40) – batch 6 –samples 24 (Worst)
3. (Script3) Try different spatial window\_sizes, samples per volume and batch.
   1. (24,24,24) – batch 6 –samples 6
   2. (24,24,24) – batch 8 –samples 6 (best)
   3. (24,24,24) – batch 6 –samples 12
   4. (24,24,24) – batch 8 –samples 12 (best)
4. (Script4) Normalisation and whitenening
   1. Whitening False, Normalisation False
   2. Whitening=False, Normalisation=True
   3. True, False
   4. True, True
   5. Whitening False, Normalisation False, normalise\_foreground\_only=True
   6. **Whitening=False, Normalisation=True (Best),** **normalise\_foreground\_only=True**
   7. True, False, normalise\_foreground\_only=True
   8. True, True, normalise\_foreground\_only=True
5. (Script5) Optimizer
   1. **Adam (Mejor)**
   2. Adagrad (Peor)
   3. Gradientdescent (malo)
   4. Momentum (Segundo major)
   5. Nesterov
6. (Script6) Activation\_function
   1. Relu (casi igual que Prelu)
   2. Relu6
   3. Elu
   4. Softplus (Segundo major)
   5. Softsign
   6. Sigmoid
   7. Tanh
   8. **Prelu (Mejor)**
   9. Selu
   10. Leakyrelu
7. (Script7) Augmentation
   1. scaling\_percentage=-1.5,1.5,0.8 (casi igual que sin augmentation)
   2. random\_flipping\_axes=0,1 (empeora)
   3. rotation\_angle= -10,10 (casi igual)
   4. do\_elastic\_deformation = True (mejora un poquito)
   5. Using all of the above (empeora)
   6. No augmentation (Mejor)
8. (Script8) Fixed learning rate without regularization
   1. 0.01 (60mil iter)

**30mil iteraciones tarda 24 horas**



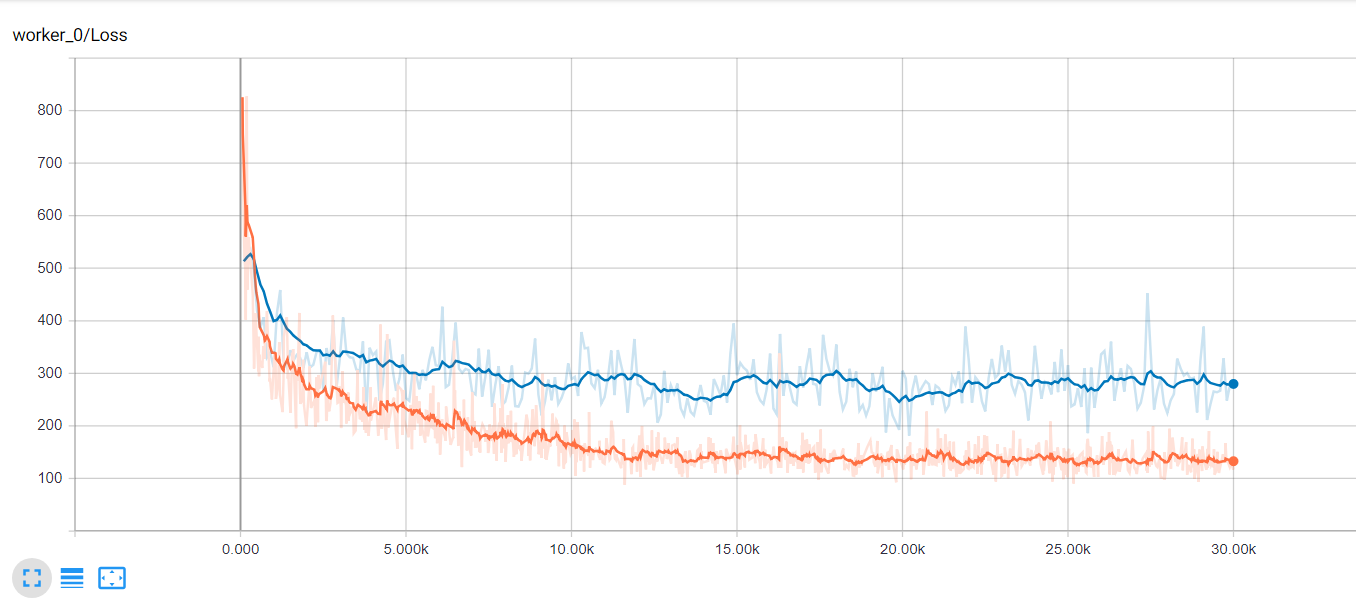
1. (Script9) Decaying learning rate without regularization
   1. 0.01 (20mil iter)
   2. 0.001 (40mil iter)
   3. 0.0001 (60mil iter)

-Total run time = 38 hours.

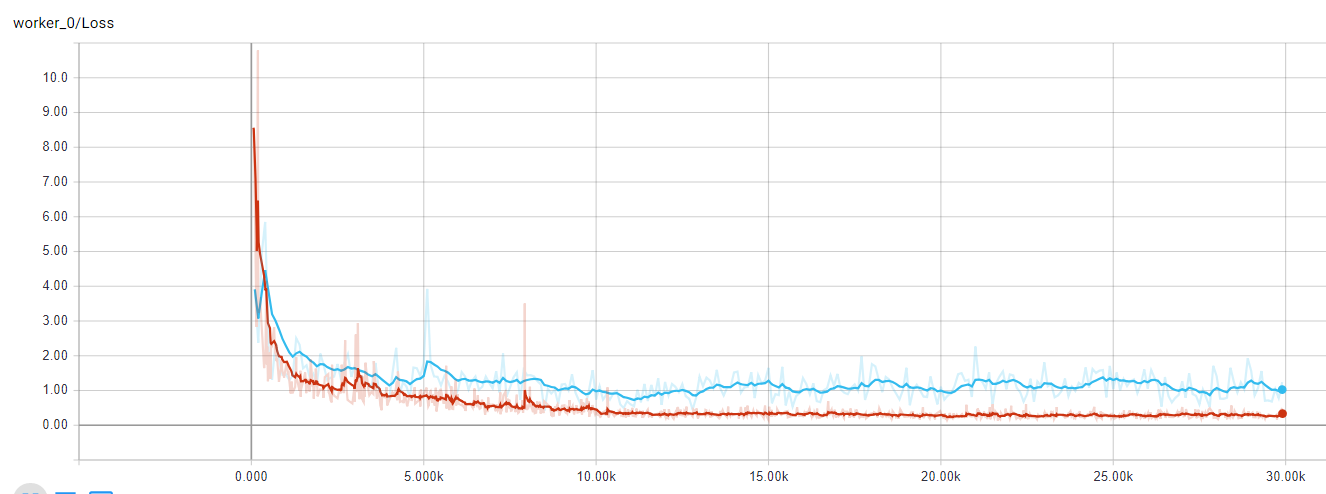


1. (Script10) Decaying learning rate without regularization
   1. 0.01 (10mil iter)
   2. 0.001 (20mil iter)
   3. 0.0001 (30mil iter)

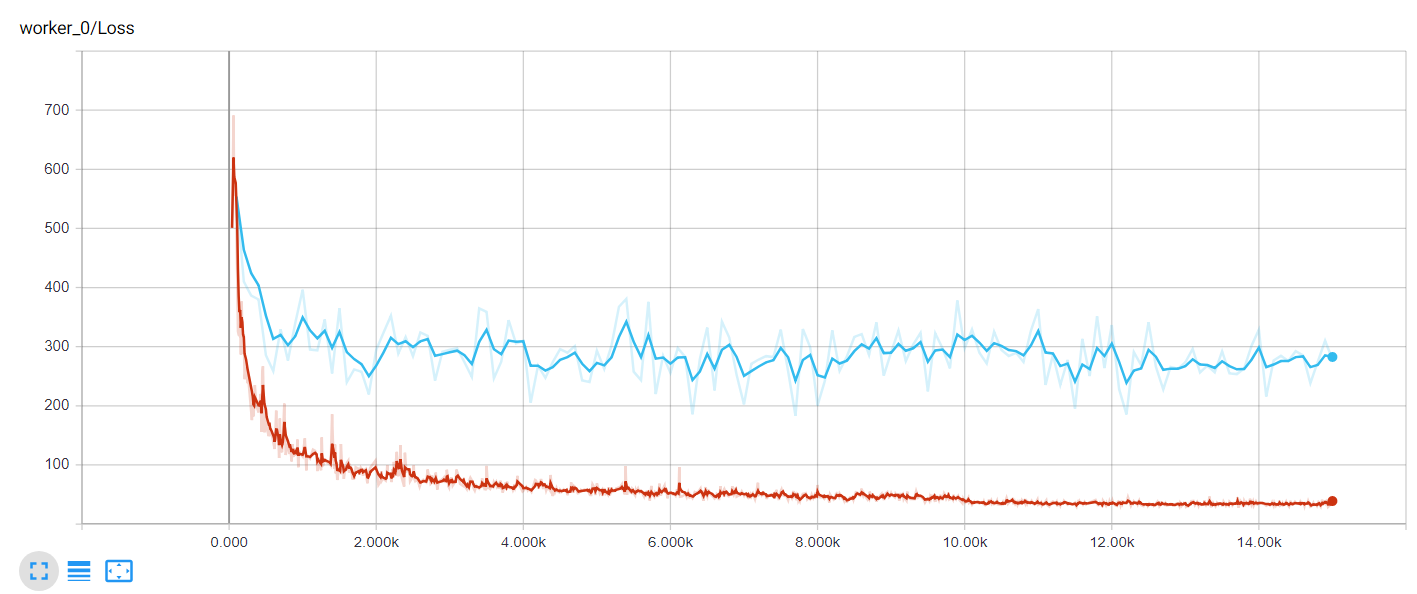
**Usar hasta 20000iter.**



1. (Script11\_L2Loss) Lo mismo que el script10 pero usando L2Loss para comparar con el artículo de (Kläser et al. 2018).



1. (Script) Overfitting 1 sample (patient\_003)
   1. 0.01 (10mil) iter, 0.001 (15mil iter)
      1. Batch size = 1 (naranja)
      2. Batch size = 6 (rojo)



>

Evaluation:

Hay que incluir sección [INFERRED] y [REGRESSION]

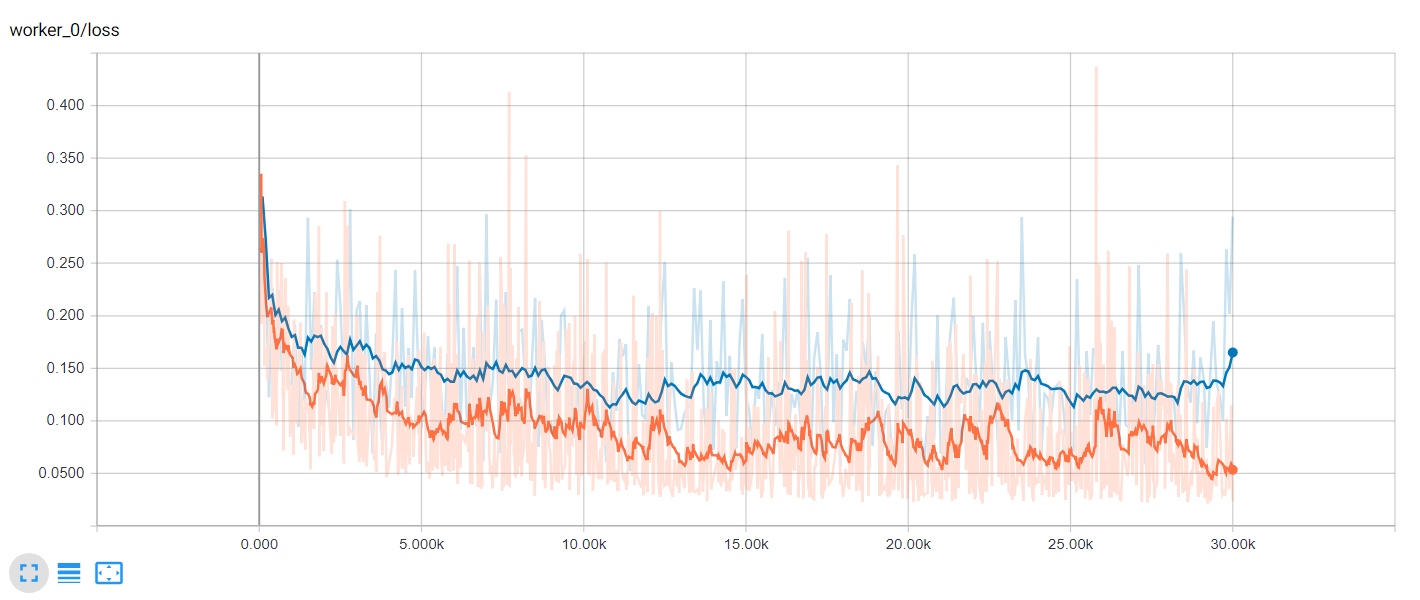
<

**Segmentation**

**Highres3dnet with weighted sampling**

1. (Script1) Decaying learning rate without regularization (mismos parámetros que en Regression)
   1. 0.01 (10mil iter)
   2. 0.001 (20mil iter)
   3. 0.0001 (30mil iter)

**Usar 20mil iter**



**Vnet with weighted sampling**

**Vnet with weighted sampling**

**No consigo hacer fine\_tuning ni para regression ni para segmentation – 28/08/2018**

**Probar usar las imágenes en .ini para que las iteraciones sean más rápidas (NO HAY DIFERENCIA con mhd, igual algunas iteraciones tardan más tiempo).**