Linear and Deformable Image Registration with 3D Convolutional Neural Networks

Stergios C. et al.

Stoyanov D. et al. (eds) Image Analysis for Moving Organ, Breast, and Thoracic Images. RAMBO 2018, BIA 2018, TIA 2018

Presented by - Kumar Shreshtha
BITS Pilani

Outline

Outline

Image Registration

- The process of establishing spatial correspondence between a source image (I_S) and a target image (I_T) to align them to the same coordinate system.

Image Registration

- The process of establishing spatial correspondence between a source image (I_S) and a target image (I_T) to align them to the same coordinate system.
- Optimization Problem:

$$\theta^* = \min_{\theta} C(I_T, T_\theta(I_S))$$

where	C	cost function
	I_T	target image
	I_S	source image
	θ	transformation parameters
	T_θ	transformation function

Image Registration

- The process of establishing spatial correspondence between a source image (I_S) and a target image (I_T) to align them to the same coordinate system.
- Optimization Problem:

$$\theta^* = \min_{\theta} C(I_T, T_\theta(I_S))$$

where
- C cost function
- I_T target image
- I_S source image
- θ transformation parameters
- T_θ transformation function

- The task is to optimize a cost function over two images by tuning the parameters of a transformation function which tries to warp the source image to target image.

Registration: Rigid vs Deformable

- Rigid/Linear Registration:
  - Uses a simple transform, uniformly applied.

Registration: Rigid vs Deformable

- Rigid/Linear Registration:
  - Uses a simple transform, uniformly applied.
  - Rotations, translations, etc.

Registration: Rigid vs Deformable

- Rigid/Linear Registration:
  - Uses a simple transform, uniformly applied.
  - Rotations, translations, etc.
- Non-Rigid/Deformable Registration:
  - Allows a non-uniform mapping between images.

Registration: Rigid vs Deformable

- Rigid/Linear Registration:
  - Uses a simple transform, uniformly applied.
  - Rotations, translations, etc.
- Non-Rigid/Deformable Registration:
  - Allows a non-uniform mapping between images.
  - Measure and/or correct small, varying discrepancies by deforming one image to match the other.

Registration: Rigid vs Deformable

- Rigid/Linear Registration:
  - Uses a simple transform, uniformly applied.
  - Rotations, translations, etc.
- Non-Rigid/Deformable Registration:
  - Allows a non-uniform mapping between images.
  - Measure and/or correct small, varying discrepancies by deforming one image to match the other.
  - Vector field/deformation field T_θ is computed from I_T to I_S.
  - Inverse warp transform I_S into I_T's coordinate system.

Deformable Registration: Clinical Relevance

- internal organs are non-rigid
- physical brain deformations during neurosurgery
- normal squishing, shifting and emptying of abdominal/pelvic organs and soft tissues
- Patient motion during image scanning
- **Lung motion during respiration**

Registration: ill-posed problem

- Well-posed problems:
  - solution exists
  - is unique
  - depends continuously on the data

Registration: ill-posed problem

- Well-posed problems:
  - solution exists
  - is unique
  - depends continuously on the data
- Ambiguity between homogeneous regions. Multiple solutions possible.(not Unique)

Registration: ill-posed problem

- Well-posed problems:
  - solution exists
  - is unique
  - depends continuously on the data
- Ambiguity between homogeneous regions. Multiple solutions possible.(not Unique)
- Solution search space is often ∞-dimensional.

Registration: Cost Function

- The cost/energy function typically comprises of two components:

Registration: Cost Function

- The cost/energy function typically comprises of two components:

$$C(I_T, T_\theta(I_S)) = \mathcal{M}(I_T, I_S \circ T_\theta) + \mathcal{R}(T_\theta)$$

where $\mathcal{M}$ quantifies level of alignment between I_T and I_S
$\mathcal{R}$ regularizes the transformation

Registration: Cost Function

- The cost/energy function typically comprises of two components:

$$C(I_T, T_\theta(I_S)) = \mathcal{M}(I_T, I_S \circ T_\theta) + \mathcal{R}(T_\theta)$$

where $\mathcal{M}$ quantifies level of alignment between I_T and I_S
$\mathcal{R}$ regularizes the transformation

- $\mathcal{R}$ aims to favor specific properties desired in the solution.

Registration: Cost Function

- The cost/energy function typically comprises of two components:

$$C(I_T, T_\theta(I_S)) = \mathcal{M}(I_T, I_S \circ T_\theta) + \mathcal{R}(T_\theta)$$

where $\mathcal{M}$ quantifies level of alignment between I_T and I_S
$\mathcal{R}$ regularizes the transformation

- $\mathcal{R}$ aims to favor specific properties desired in the solution.
- Helps tackle the ill-posedness of registration problem.

Registration: Cost Function

- The cost/energy function typically comprises of two components:

$$C(I_T, T_\theta(I_S)) = \mathcal{M}(I_T, I_S \circ T_\theta) + \mathcal{R}(T_\theta)$$

where $\mathcal{M}$ quantifies level of alignment between I_T and I_S
$\mathcal{R}$ regularizes the transformation

- $\mathcal{R}$ aims to favor specific properties desired in the solution.
- Helps tackle the ill-posedness of registration problem.
- Adding a regularization term can provide **provable uniqueness** and a **computable subspace**.

Registration: Ingredients

- Image Registration Algorithm involves three main components:

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$
  - optimization method

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$
  - optimization method
- The dependency of the registration result on the optimization strategy follows from the fact that image registration is inherently ill-posed.

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$
  - optimization method
- The dependency of the registration result on the optimization strategy follows from the fact that image registration is inherently ill-posed.
- Devising each component so that the requirements of the registration algorithm are met is a demanding process.

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$
  - optimization method
- The dependency of the registration result on the optimization strategy follows from the fact that image registration is inherently ill-posed.
- Devising each component so that the requirements of the registration algorithm are met is a demanding process.
- Modeling Complex Equations?

Registration: Ingredients

- Image Registration Algorithm involves three main components:
  - deformation model - T_θ
  - objective function - $C(I_T, T_\theta(I_S))$
  - optimization method
- The dependency of the registration result on the optimization strategy follows from the fact that image registration is inherently ill-posed.
- Devising each component so that the requirements of the registration algorithm are met is a demanding process.
- Modeling Complex Equations?
- **Neural Networks!**

Existing Methods

- Drawbacks of state-of-the-art methods:
  - Existing methods are often biased towards the linear component
  - Sensitive to application setting
  - involve multiple hyper-parameters
  - computationally expensive

Existing Methods

- Drawbacks of state-of-the-art methods:
  - Existing methods are often biased towards the linear component
  - Sensitive to application setting
  - involve multiple hyper-parameters
  - computationally expensive
- Limitations of Deep learning based methods:
  - dependency on linear component of the transformation
  - **dependency on ground truth displacement (supervised learning)**

Contributions of Stergios C. et al.

- coupling linear and deformable registration within a single architecture.
- independent of different similarity metrics
- reduced computational time
- associating with clinical data to classify patients with interstitial lung disease (ILD).

Outline

Dataset

- MRI exams of 41 patients were acquired(29 with systemic sclerosis and 12 healthy)
- Lung fields were annotated for 82 images and were labeled as healthy/not healthy.

Dataset

- MRI exams of 41 patients were acquired(29 with systemic sclerosis and 12 healthy)
- Lung fields were annotated for 82 images and were labeled as healthy/not healthy.
- Pre-processing:
  - image intensity values clipped between the range [0,1300] and normalized to [0,1].
  - images were scaled down by a factor of 2/3 using cubic interpolation to a final size of 64x192x192.
  - 28 patients data were selected for training using random sampling and remaining 13 were used for validation.

Network Architecture

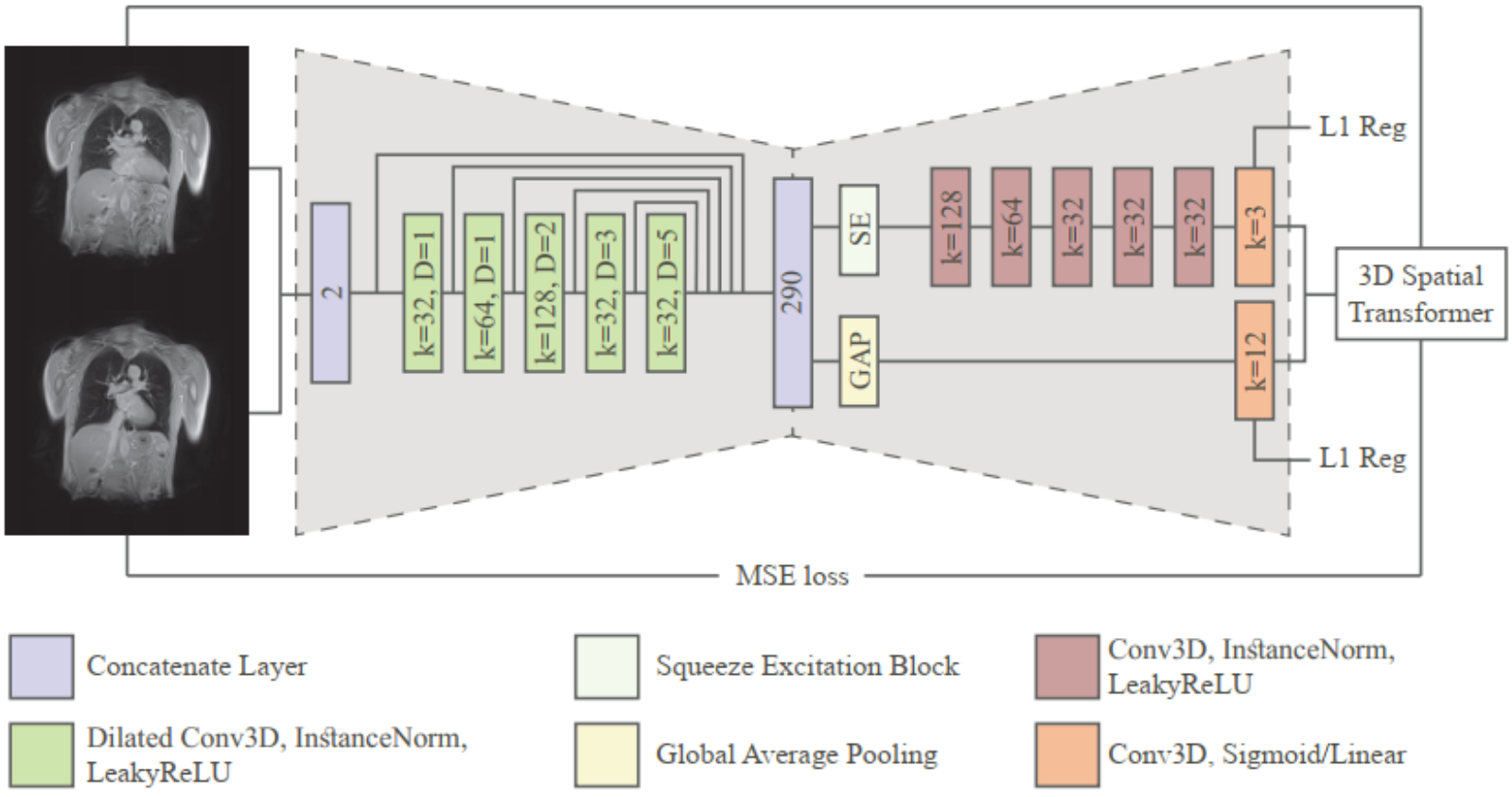


Figure 1: Overview of Network Architecture

Network Architecture: PyTorch

```
class Register3d(nn.Module):
    def __init__(self):
        super(Register3d,self).__init__()
        self.encoder = Encoder()
        self.d_decoder = DRB()
        self.a_decoder = LRB()
        self.transformer = Transformer3d()

    def forward(self,source,target,linear = False):
        x = self.encoder(source,target)
        d1 = self.d_decoder(x)
        if not linear:
            t = self.transformer(d1)
        else:
            d2 = self.a_decoder(x)
            t = self.transformer(d1,d2)
        return (t,d1,d2)
```

Network Architecture: Encoder

- takes as input the source and target image concatenated along channels axis.
- Consists of 5 layers of 3x3x3 dilated convolutions each followed by instance normalization and LeakyReLU.
- the outputs of each layer are finally concatenated to provide multi-resolution feature merging which is then fed into the 2 branches of the decoder.

Encoder: Insights

- why dilated convolution?
  - provides greater receptive field to compensate for no max-pooling.
  - larger filters require more trainable parameters.

Encoder: Insights

- why dilated convolution?
  - provides greater receptive field to compensate for no max-pooling.
  - larger filters require more trainable parameters.
- gridding problem: as zeros are padded between two pixels in a convolutional kernel, the receptive field of this kernel only covers an area with checkerboard patterns - only locations with non-zero values are sampled, losing some neighboring information.

Encoder: Insights

- why dilated convolution?
  - provides greater receptive field to compensate for no max-pooling.
  - larger filters require more trainable parameters.
- gridding problem: as zeros are padded between two pixels in a convolutional kernel, the receptive field of this kernel only covers an area with checkerboard patterns - only locations with non-zero values are sampled, losing some neighboring information.
- dilation rate increases as Fibonacci sequence (1,1,2,3,5) as opposed to exponentially. This mitigates the gridding problem by providing a less steep dilation rate increase and thus denser sampling.

Encoder: Insights (Contd.)

- Instance normalization
  - provides invariance to intensity and contrast shift
  - simplifies learning process
  - could mitigate problems caused by difference in imaging devices.

Encoder: Insights (Contd.)

- Instance normalization
  - provides invariance to intensity and contrast shift
  - simplifies learning process
  - could mitigate problems caused by difference in imaging devices.
- multi-resolution feature merging
  - permits the aggregation of features from all different scales and levels of abstraction
  - facilitates the flow of gradients through the network and therefore allows faster training.

Encoder: PyTorch

```
class Encoder(nn.Module):
    def __init__(self):
        super(Encoder,self).__init__()
        self.conv1 = nn.Conv3d(6,32,3, padding = 1)
        self.conv2 = nn.Conv3d(32,64,3,padding = 1)
        self.conv3 = nn.Conv3d(64,128,3,padding = 2,dilation = 2)
        self.conv4 = nn.Conv3d(128,32,3,padding = 3,dilation = 3)
        self.conv5 = nn.Conv3d(128,32,3,padding = 5,dilation = 5)
    def forward(self,source,target):
        x0 = torch.cat(source,target,dim=1)
        x1 = F.leaky_relu(F.instance_norm(self.conv1(x0)))
        x2 = F.leaky_relu(F.instance_norm(self.conv1(x1)))
        x3 = F.leaky_relu(F.instance_norm(self.conv1(x2)))
        x4 = F.leaky_relu(F.instance_norm(self.conv1(x3)))
        x5 = F.leaky_relu(F.instance_norm(self.conv1(x4)))
        return torch.cat((x0,x1,x2,x3,x4,x5),dim=1)
```

Network Architecture: Decoder

- comprises of two branches each of which produces a sampling grid
  - for Linear Registration
  - for Deformable Registration

  respectively.

Network Architecture: Decoder

- comprises of two branches each of which produces a sampling grid
  - for Linear Registration
  - for Deformable Registration

  respectively.
- **Linear Registration Branch (LRB)** comprises of a global average pooling layer followed by 1x1x1 3D convolutions with linear activation to produce 12(3x4) activation values which acts as the affine transformation matrix. This affine matrix is then used to compute a sampling grid for linear registration.

Network Architecture: Decoder

- comprises of two branches each of which produces a sampling grid
  - for Linear Registration
  - for Deformable Registration

  respectively.
- **Linear Registration Branch (LRB)** comprises of a global average pooling layer followed by 1x1x1 3D convolutions with linear activation to produce 12(3x4) activation values which acts as the affine transformation matrix. This affine matrix is then used to compute a sampling grid for linear registration.
- **Deformable Registration Branch (DRB)** comprises of a squeeze and excitation block followed by multiple 3x3x3 convolutions each followed by instance norm and LeakyReLU. The final layer has sigmoid activation and its activations are multiplied by 2 to get a spatial gradient sampling grid.

Decoder: PyTorch

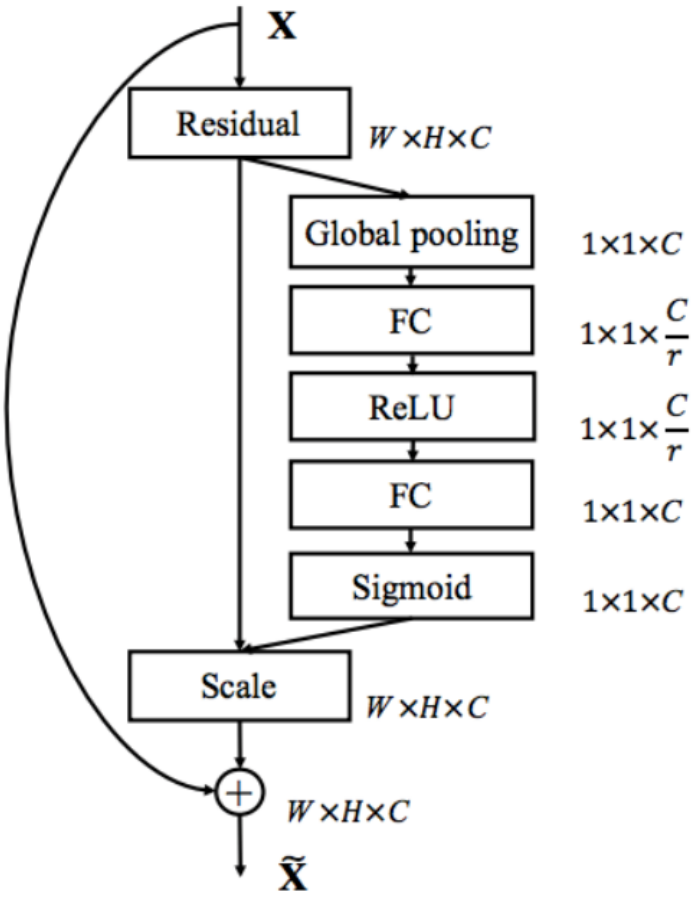
```
class LRB(nn.Module):
    def __init__(self):
        super(LRB,self).__init__()
        self.conv1 = nn.Conv3d(290,12,1)
    def forward(self,inputs):
        x = self.conv1(F.adaptive_avg_pool3d(inputs,(None,None,None,1,1))).view(-1,3,4)
        return x
```

```
class SqueezeExcitation(nn.Module):
    def __init__(self,channel,reduction=16):
        super(SqueezeExcitation,self).__init__()
        self.conv1 = nn.Conv3d(channel,channel//reduction,1)
        self.conv2 = nn.Conv3d(channel//reduction,channel,1)
    def forward(self,x):
        return x*F.sigmoid(self.conv2(F.relu(self.conv1(F.adaptive_avg_pool3d(x,(None,None,1,1))))))
```

Decoder: PyTorch

```
class DRB(nn.Module):
    def __init__(self):
        super(DRB,self).__init__()
        self.se = SqueezeExcitation(290)
        self.conv1 = nn.Conv3d(290,128,3,padding = 1)
        self.conv2 = nn.Conv3d(128,64,3,padding = 1)
        self.conv3 = nn.Conv3d(64,32,3,padding = 1)
        self.conv4 = nn.Conv3d(32,32,3,padding = 1)
        self.conv5 = nn.Conv3d(32,32,3,padding = 1)
        self.conv6 = nn.Conv3d(32,3,3,padding = 1)
    def forward(self,inputs):
        x = self.se(inputs)
        x = F.leaky_relu(F.instance_norm(self.conv1(x)))
        x = F.leaky_relu(F.instance_norm(self.conv2(x)))
        x = F.leaky_relu(F.instance_norm(self.conv3(x)))
        x = F.leaky_relu(F.instance_norm(self.conv4(x)))
        x = F.leaky_relu(F.instance_norm(self.conv5(x)))
        x = 2*F.sigmoid(self.conv6(x))
        return x
```

Deformable Registration Branch (DRB) : Insights



- Squeeze and Excitation Block:
  - in a normal convolution operation all channels contribute equally to the output feature map.
  - SE-block allows to add an adaptive scale to control the contribution of each input channel to the output feature map.

Deformable Registration Branch (DRB) : Insights

- The final output of the DRB is a 3 channel 3d-feature map of same spatial dimensions as the target image with values in range [0,2].

Deformable Registration Branch (DRB) : Insights

- The final output of the DRB is a 3 channel 3d-feature map of same spatial dimensions as the target image with values in range [0,2].
- Thus, for each voxel we have 3 values between [0,2] representing the spatial gradient along each of the three axes (x,y,z).

Deformable Registration Branch (DRB) : Insights

- The final output of the DRB is a 3 channel 3d-feature map of same spatial dimensions as the target image with values in range [0,2].
- Thus, for each voxel we have 3 values between [0,2] representing the spatial gradient along each of the three axes (x,y,z).
- The spatial gradient gives a measure of displacement of consecutive pixels.

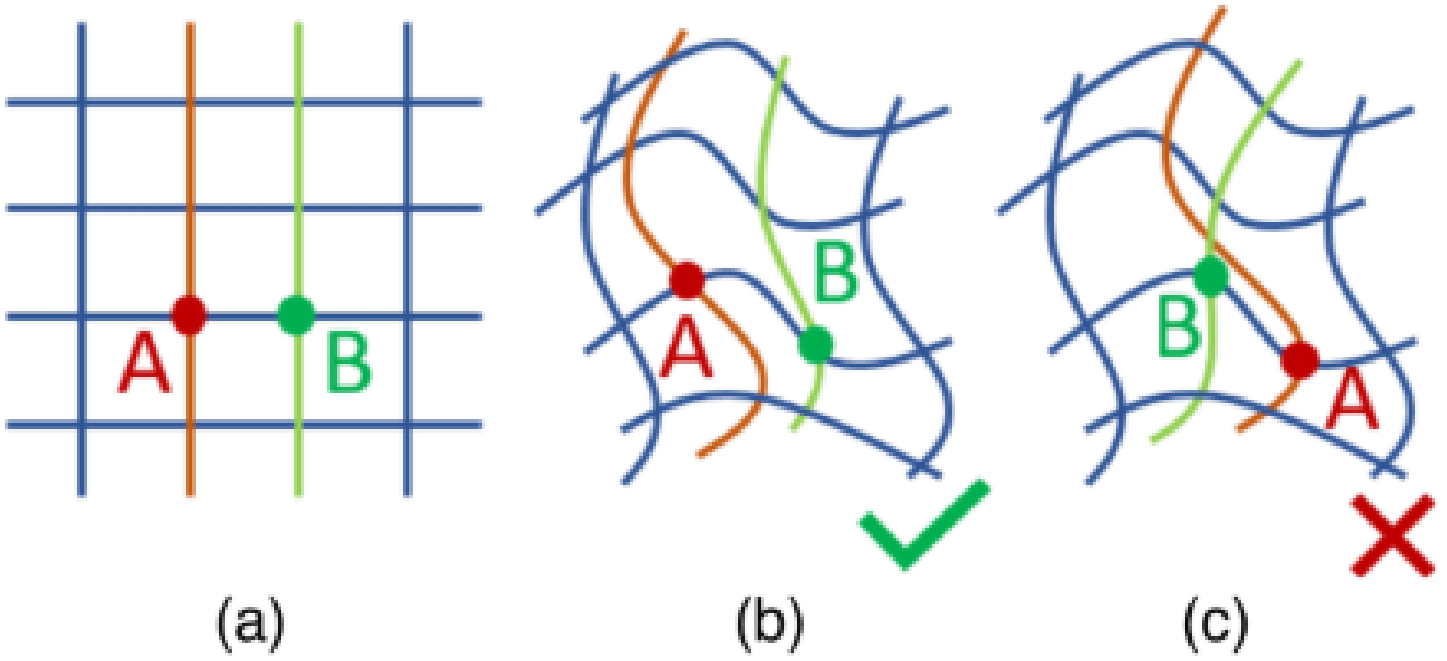
Deformable Registration Branch (DRB) : Insights

- The final output of the DRB is a 3 channel 3d-feature map of same spatial dimensions as the target image with values in range [0,2].
- Thus, for each voxel we have 3 values between [0,2] representing the spatial gradient along each of the three axes (x,y,z).
- The spatial gradient gives a measure of displacement of consecutive pixels.
- limiting the maximum value of gradients to two allows for control of maximum displacements among consecutive pixels.

Deformable Registration Branch (DRB) : Insights

- The final output of the DRB is a 3 channel 3d-feature map of same spatial dimensions as the target image with values in range [0,2].
- Thus, for each voxel we have 3 values between [0,2] representing the spatial gradient along each of the three axes (x,y,z).
- The spatial gradient gives a measure of displacement of consecutive pixels.
- limiting the maximum value of gradients to two allows for control of maximum displacements among consecutive pixels.
- such an approach ensures generation of smooth deformations that avoid self-crossing.

Deformable Registration Branch (DRB) : Insights



Non-negative values for spatial gradients ensure no **self-crossing**.

Deformable Registration Branch (DRB) : Insights

- A spatial gradient value of 1 indicates there is no change in the distance between the consecutive pixels along the given axis.
- A value less than 1 indicate the distance between the pixels will decrease along the given axis.
- A value greater than 1 indicate the distance between the pixels will increase along the given axis.

Deformable Registration Branch (DRB) : Insights

- A spatial gradient value of 1 indicates there is no change in the distance between the consecutive pixels along the given axis.
- A value less than 1 indicate the distance between the pixels will decrease along the given axis.
- A value greater than 1 indicate the distance between the pixels will increase along the given axis.

Generating sampling grid from spatial gradients

$$[G(p)]_d = \sum_{i=0}^{p} \phi_{id}$$

with $[G(p)]_d$ d-component of sampling grid G at voxel p

ϕ_{id} d-component of spatial gradient ϕ at voxel i

Linear Registration Branch (LRB)

- once the affine transformation matrix is calculated using the above procedure we can generate the sampling grid for each voxel in the deformed image using the following formula:

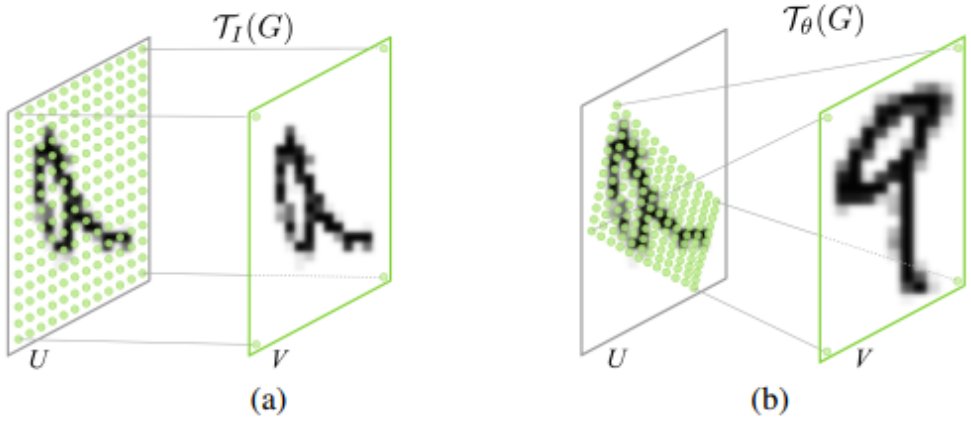
Generating sampling grid from affine matrix

$$\begin{pmatrix} x_i^s \\ y_i^s \\ z_i^s \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} & \theta_{14} \\ \theta_{21} & \theta_{22} & \theta_{23} & \theta_{24} \\ \theta_{31} & \theta_{32} & \theta_{33} & \theta_{34} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ z_i^t \\ 1 \end{pmatrix}$$

3D Spatial Transformer

- the 3D transformer layer takes as input the source(moving) image (S), the linear(G_A) and deformable (G_N) sampling grids and performs the sampling operation $\mathcal{W}$ to produce the deformed image D.
- first the deformable grid is applied and to its output the linear sampling grid is applied:

$$\mathcal{W}(S, G) = \mathcal{W}(\mathcal{W}(S, G_N), G_A)$$



3D Spatial Transformer

- The warping operation ($\mathcal{W}$) must be differentiable with respect to the input image (S) and the deformation(G) in order to allow back-propagation training.

3D Spatial Transformer

- The warping operation ($\mathcal{W}$) must be differentiable with respect to the input image (S) and the deformation(G) in order to allow back-propagation training.
- The warping function is thus defined as a trilinear interpolation sampling as follows:

$$\mathcal{W}(S, G)(\mathbf{p}) = \sum_{\mathbf{q}} S(\mathbf{q}) \prod_{d} max(0, 1 - |[G(\mathbf{p})]_d - \mathbf{q}_d|)$$

Training

Loss Function

$$Loss = ||R - \mathcal{W}(S, G)||^2 + \alpha||A - A_I||_1 + \beta||\Phi - \Phi_I||_1$$

Training

Loss Function

$$Loss = ||R - \mathcal{W}(S, G)||^2 + \alpha||A - A_I||_1 + \beta||\Phi - \Phi_I||_1$$

- Why regularization?

Training

Loss Function

$$Loss = ||R - \mathcal{W}(S, G)||^2 + \alpha||A - A_I||_1 + \beta||\Phi - \Phi_I||_1$$

- Why regularization?
  - Without L1 regularization on A high reconstruction error may occur as the network might get stuck in a local minimum where it aligns only high level features.

Training

Loss Function

$$Loss = ||R - \mathcal{W}(S, G)||^2 + \alpha||A - A_I||_1 + \beta||\Phi - \Phi_I||_1$$

- Why regularization?
  - Without L1 regularization on A high reconstruction error may occur as the network might get stuck in a local minimum where it aligns only high level features.
  - Without the smoothness regularizer on ϕ, spatial gradients decoder network can predict very non-smooth grids which makes it prone to fall in local minimum.

Training (Contd.)

hyper-parameters:

- **optimizer:** Adam
- **initial learning rate:** 10^{-3}
- **reduce LR on plateau:**
  - patience: 50 epochs
  - factor: 0.9
- **early stopping patience :** 100 epochs
- **batch size:** 2
- **regularizers:** $\alpha = \beta = 10^{-6}$

Outline

Evaluation

- Dice Coefficient Metric
  - measured on lung masks after source deformation.
- Average error over all axes of predicted locations for 11 manually annotated landmark points was also calculated.
- Compared with two different state-of-the-art methods
  - Symmetric Normalization(SyN)
  - deformable method with variety of similarity metrics (MI,NCC,DWM)

Results

- Two different types of test:
  - Inhale-Exhale (13 validation pairs)
  - All combinations (169 validation pairs)

Method	Inhale-Exhale	All Combinations	Time/subject(s)
Unregistered	75.620±10.89	57.22±12.90	-
Deformable with NCC	84.25±6.89	76.10±7.92	1(GPU)
Deformable with DWM	88.63±4.67	75.92±8.81	2(GPU)
Deformable with MI	88.86±5.13	76.33±8.74	2(GPU)
Deformable with all above	88.81±5.85	78.71±8.56	2(GPU)
SyN	83.86±6.04	-	2500(CPU)
Proposed w/o Affine	91.28±2.47	81.75±7.88	0.5(GPU)
Proposed	**91.48±2.33**	**82.34±7.68**	0.5(GPU)

Table 1: Dice Coefficient Score(%) calculated over the deformed lung masks and the ground truth

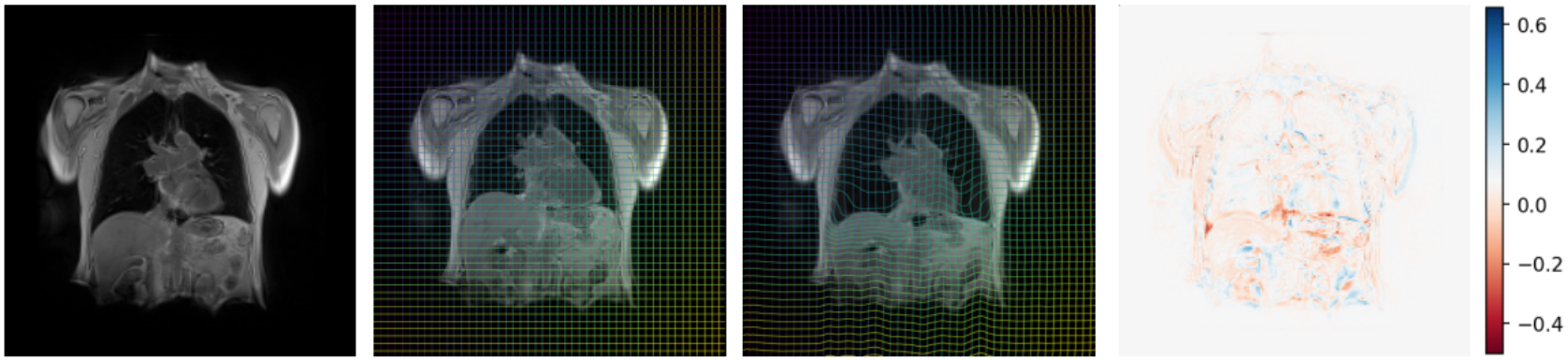
Results (Contd.)

Method	dx	dy	dz	ds
Inter-observer	1.664	2.545	1.555	3.905
Deformable with NCC,DWM and MI	1.855	3.169	2.229	4.699
Proposed w/o Affine	2.014	2.947	1.858	4.569
Proposed	1.793	2.904	1.822	4.358

Table 2: Errors measured as average euclidian distances between estimated landmark locations and ground truth marked by two medical experts.

Results: Analysis

Majority of the errors occur at boundaries, indicating that large local displacements are not captured



(a) Reference image (b) Moving image (c) Deformed image (d) Difference

Figure 2: Registration generated by the proposed architecture

Evaluation of Clinical Relevance

- a small classifier is trained for assessing clinical relevance.
- residual deformation ($G_\delta = G - G_I$) indicating voxel displacement is fed as input to a shallow neural network which classifies patients as healthy/unhealthy.
- binary crossentropy was used as loss function and initial learning rate 10^{-4}, which was halved every 50 epochs.
- 5 models were trained in parallel for 5-fold cross validation and an accuracy of 84.62 % is reported.

Evaluation of Clinical Relevance: Model

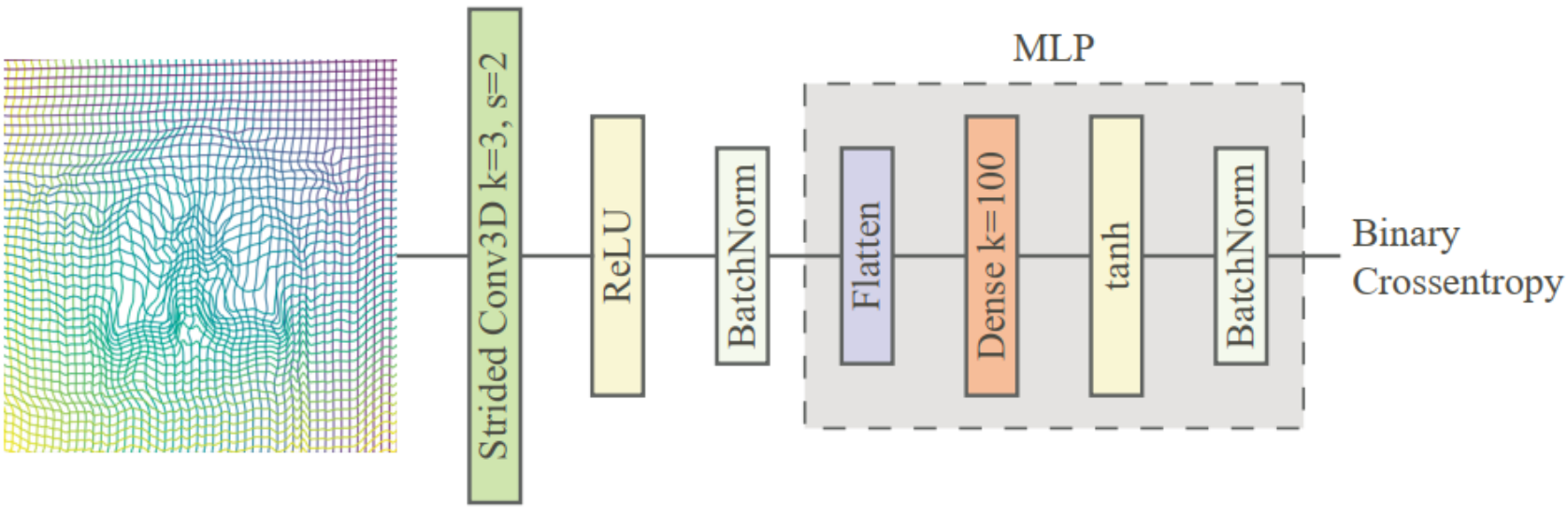


Figure 3: ILD Classifier

Outline

Conclusion

- The paper presents an unsupervised end-to-end deep learning model for image registration using an optimal transformation between pair of images.
- the model is modular with respect to similarity functions and nature of transformation
- promising results compared to other unsupervised methods
- generates deformations with no self-crossings
- computationally efficient due to GPU implementation
- **Future Scope:** Evaluation of the proposed model on registration of other modalities and other organs.