SACB-Net

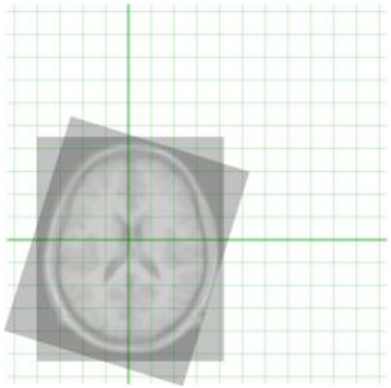
Spatial-awareness Convolutions for Medical Image Registration

Presented by Akindu Kalhan

Motivation

The Medical Image Registration Problem

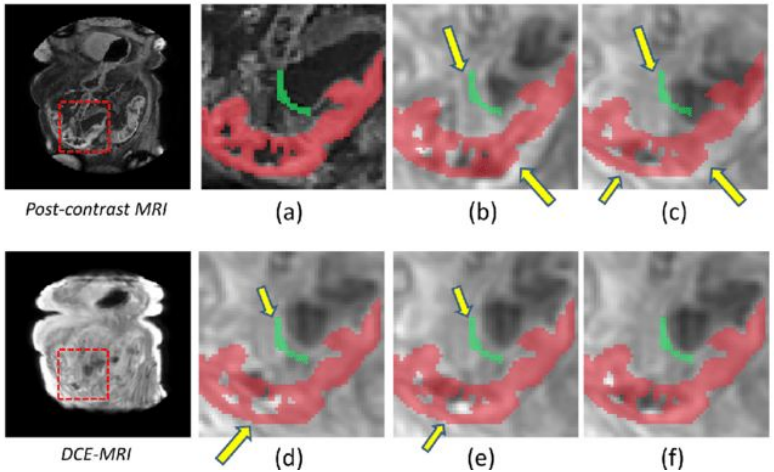
- Image Registration is the process of transforming different sets of data into one coordinate system.
- Medical Image Registration aligns a moving image to a fixed image so that corresponding anatomical structure match.
- Essential in brain MRI, abdominal CT etc.



Registration of two MRI images of the brain

Limitations of Vanilla 3D CNNs

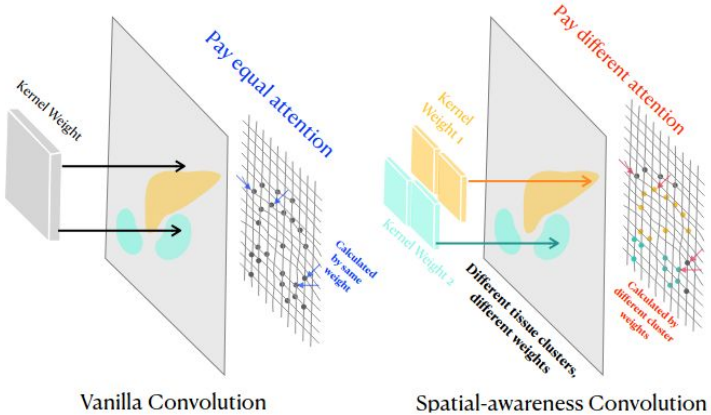
- Use shared convolution kernels across entire volume.
- Assume all regions deform similarly, which is not true for tissue.
- Fails in areas where deformation varies significantly



A large bowel deformation in a post-contrast MR image

Key Challenge

- Different anatomical regions require different filters.
- CNN filters should ***adapt*** depending on where in the anatomy the filter is applied.
- This paper introduces a solution to address this issue: “Spatial-awareness Convolutions”



Related Work

Classical Registration

- SyN
- Demons
- LDDMM

Pros

- Accurate

Cons

- Slow
- Iterative optimization per image pair

Learning-based Regi.

- VoxelMorph
- PRNet++
- Fourier-Net

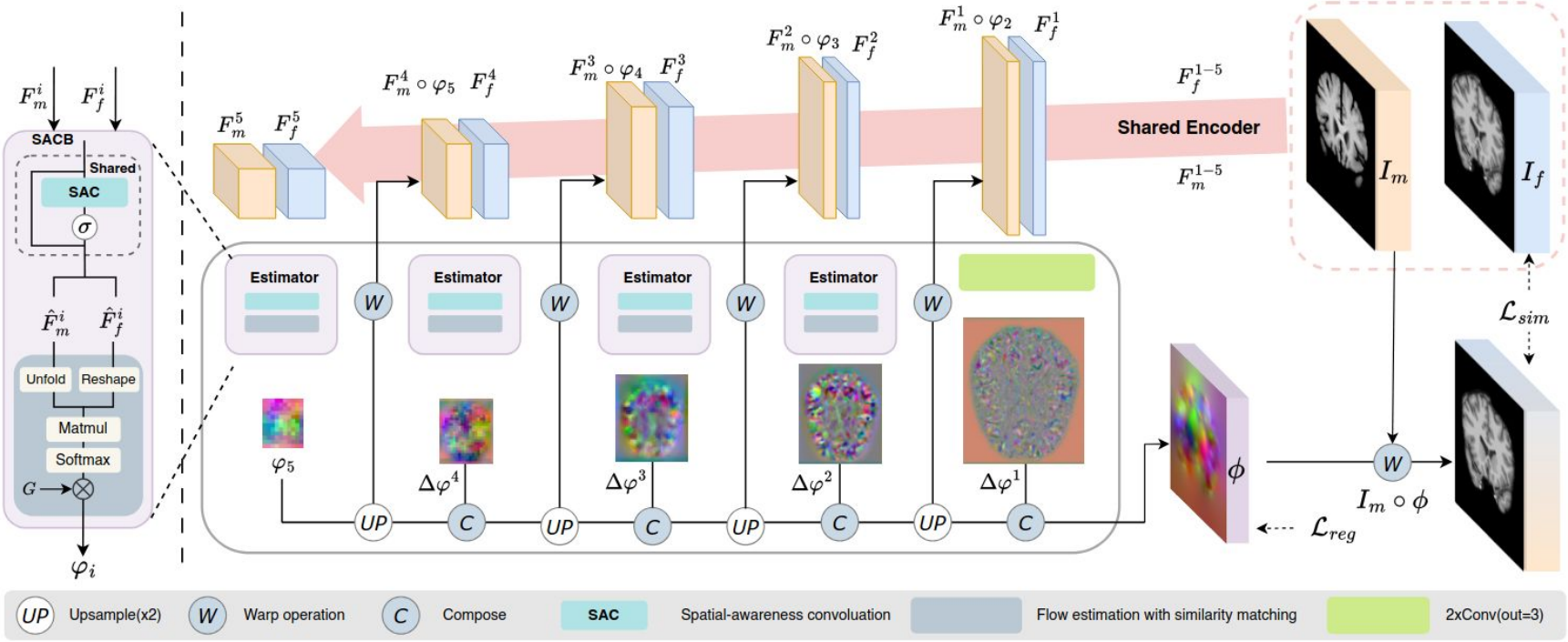
Pros

- Fast inference

Cons

- Uses *fixed* convolutional kernels
- Lack anatomical adaptiveness

SACB-Net Architecture



SACB-Net Architecture

- SACB-Net performs deformable image registration using a 5-level coarse-to-fine pyramid with SAC Blocks and feature matching flow estimation.
- Pipeline steps

    1. Shared Encoder
    2. Coarse to Fine Registration
    3. SAC Block
    4. Similarity Matching Flow Estimator
    5. Flow Composition
    6. Final 3D Deformation Field

1. Shared Encoder

- Extracts hierarchical feature maps from both the moving image Im and the fixed image If. Both the images are passed through the same CNN encoder.
- The encoder outputs 5 levels of features.

  Fixed Image Features = F^1, F^2, F^3, F^4, F^5

  Moving Image Features = M^1, M^2, M^3, M^4, M^5

- Each deeper level has a lower spatial resolution and higher number of channels.

Shared Encoder

- Each convolutional layer has a 3D convolutional operation, instance normalization (to stabilize training) and LeakyReLU activation with 0.1 slope (for non-linearities). After, the conv. layer it is followed by an average pooling operation with a kernel size of 2.
- Why 5 levels?

  Deeper layers observe a larger receptive field.

Level	Resolution	Channels	Meaning
L5	1/16	many	global layout
L4	1/8	more	organ-level structure
L3	1/4	more	anatomical regions
L2	1/2	more	shape outlines
L1	full	small	edges, fine texture

Shared Encoder

- Why do we need a Shared Encoder?

  This ensures that the features exists in the same latent space. This is essential for similarity matching.

  And this makes sure that the network is not biased towards either of the images.

2. Coarse to Fine Registration

- The main purpose is to estimate the deformation field in a gradual manner, starting from the large global motions and then refining the deformation field with local adjustments.
- Process

At the **coarsest scale (level 5)**:

- Estimate the broad alignment:

$$\phi_5 = \text{FlowEstimator}(M^5, F^5)$$

Then for each finer level i:

1. **Upsample the previous flow**

$$\phi_i^{(\text{up})} = \text{UP}(\phi_{i+1})$$

2. **Warp the moving feature**

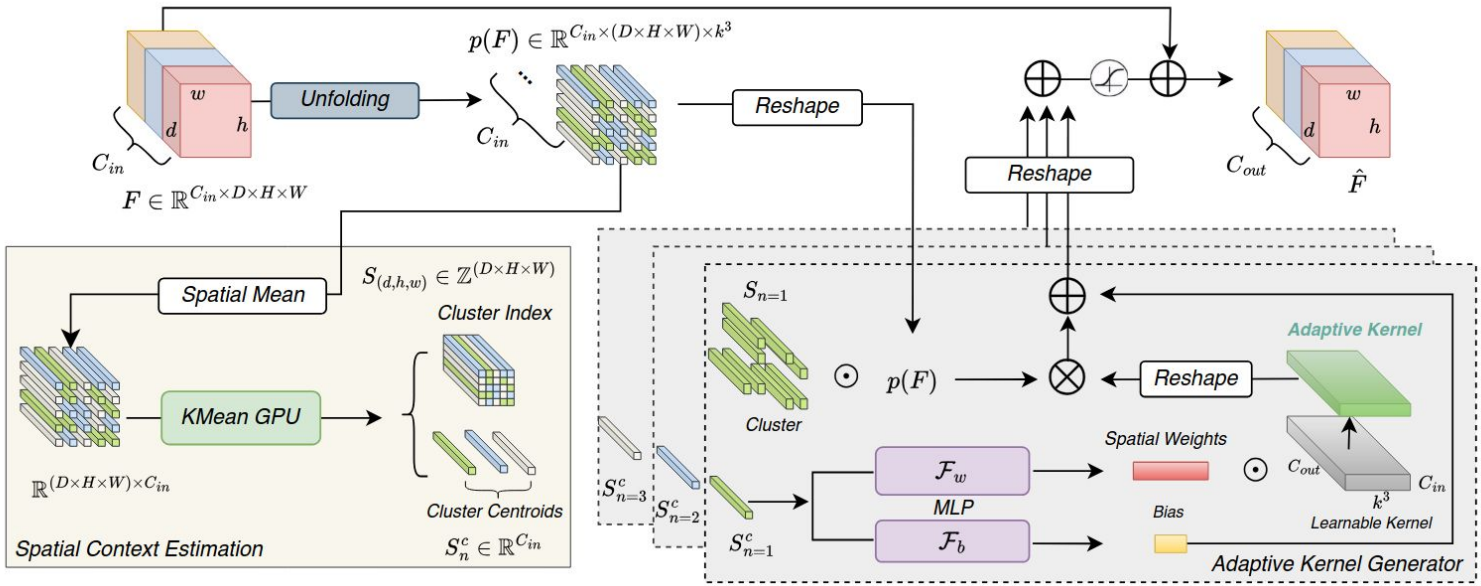
$$\tilde{M}^i = M^i \circ \phi_i^{(\text{up})}$$

3. **Compute residual flow**

$$\Delta\phi_i = \text{FlowEstimator}(\tilde{M}^i, F^i)$$

3. SAC Block

- Here we need to enable spatially adaptive filters so that, different tissue regions have different convolutional kernels.



SAC Block

- A SAC Block has three main steps:

    (a). Feature Clustering

First, the feature map F is unfolded into local patches to capture neighborhood information using sliding windows of size *k.* Next, we apply a spatial mean to reduce its spatial dimensions. And finally we apply kMeans clustering. Each spatial location will be assigned a cluster ID.

SAC Block

(b). Kernel Generation by MLP

Each cluster produces a custom convolutional kernel:

$$W_{C(x)},\ b_{C(x)} = \mathrm{MLP}(c_{\mathrm{mean}})$$

SAC Block

(c). Spatially Adaptive Convolution

At each spatial location x:

$$\hat{F}(x) = W_{C(x)} * F(x) + b_{C(x)}$$

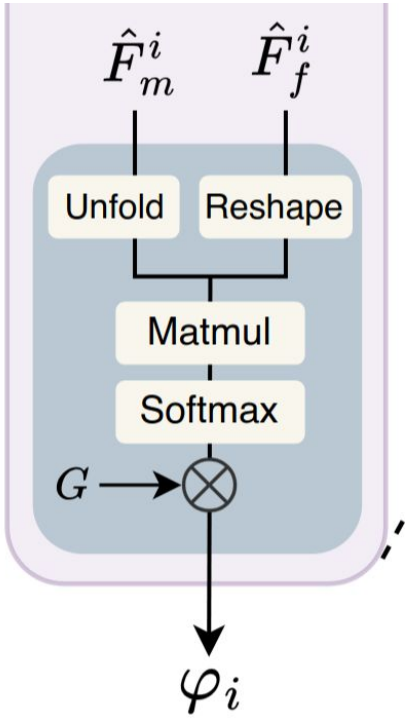
Adding Residual Connections:

$$\hat{F} = F + \sigma(\mathrm{SACB}(F))$$

4. Similarity Matching Flow Estimator

- This block computes the displacement vectors.
- Steps:

1. First, we take the moving feature map and apply unfold operation on it. The unfold operation extracts 3D patches around every location. The unfold operation turns the feature map into a matrix shape.
2. The fixed feature map is reshaped to match the patch vector size
3. Next, we take the dot product and computes a similarity score for each candidate offset v.
4. And, then we apply a softmax function to convert the similarity scores to probabilities.
5. Finally, the weighted average is taken to produce the final displacement.

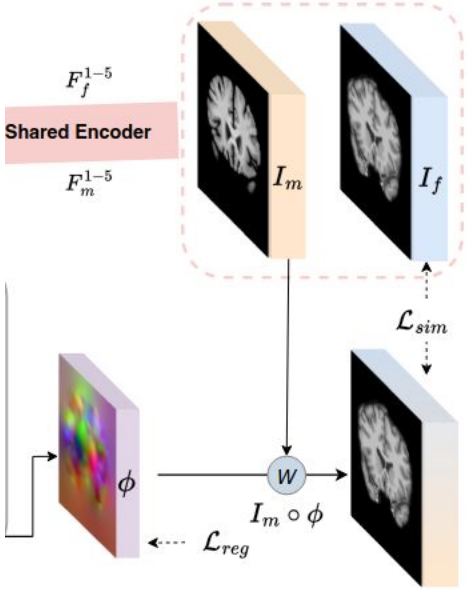


5. Flow Composition

- In here, it combine flows from coarse and fine level into a single deformation field. Coarse flow captures global alignment.
- At each level, we warp the previous flow by the new residual.
- The estimator compares the fixed features with the now warped moving image features and calculates a new flow update ($\Delta\varphi$).
- Compose (C): The new flow is composed (combined) with the upsampled flow from the previous level.
- Upsample (UP): The combined flow is upsampled by x2 to serve as the initial for the next step

6. Final 3D Deformation Field

The final displacement field Φ is produced at full resolution.



Loss Function

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{sim}(I_m \circ (\boldsymbol{\phi}(\boldsymbol{\theta}) + \mathrm{Id}), I_f) + \lambda \mathcal{L}_{reg}$$

- The regularization loss is computed by taking the L2-norm of the gradient of the deformation field.
- Normalized Cross-Correlation (NCC) is used to measure the similarity.
- The goal is to learn the optimal parameters θ.

Results

- Best Dice scores across IXI, LPBA40, and Abdomen CT.
- Lower HD95/ASSD than VoxelMorph, TransMorph, RDN, etc.
- Sharper anatomical alignment in qualitative examples.
- SACB adds +1–3% Dice improvement through adaptive kernels.
- Works well on both brain MRI and abdominal CT tasks.
- Stable, smooth deformation fields with fewer foldings.

Results

Table 1. Registration performance comparison on the IXI and LPBA datasets.

Method	**IXI** (30 ROIs)				**LPBA** (54 ROIs)				# Param
	Dice↑	HD95↓	ASSD↓	$\lvert J\rvert_{<0}\%$↓	Dice↑	HD95↓	ASSD↓	$\lvert J\rvert_{<0}\%$↓	
Affine	0.386±0.195	6.479±0.666	2.445±0.280	-	0.525±0.047	8.039±0.861	2.586±0.350	-	-
SyN [3]	0.645±0.152	6.394±1.048	1.551±0.286	<0.0001	0.707±0.016	6.254±0.444	1.479±0.131	<0.0001	-
VM-1 [5]	0.729±0.129	3.798±0.757	0.937±0.162	1.590±0.339	0.664±0.025	6.873±0.654	1.717±0.200	0.649±0.261	0.27M
VM-2 [5]	0.732±0.123	3.723±0.680	0.926±0.158	1.522±0.336	0.669±0.025	6.847±0.659	1.698±0.200	0.591±0.242	0.30M
NCA-Morph [33]	0.753±0.136	3.109±0.525	0.796±0.121	0.506±0.190	0.679±0.023	6.666±0.634	1.631±0.188	0.264±0.121	0.37M
TransMorph [10]	0.754±0.124	3.543±0.721	0.862±0.168	1.579±0.328	0.695±0.022	6.564±0.619	1.559±0.182	0.474±0.176	46.77M
LKU [20]	0.765±0.129	2.967±0.494	0.757±0.114	0.109±0.054	0.706±0.032	6.452±0.951	1.603±0.250	0.594±0.203	2.09M
B-Spline-Diff [32]	0.742±0.128	3.256±0.538	0.832±0.117	<0.0001	0.665±0.023	6.792±0.622	1.713±0.186	**0.0±0.0**	0.27M
Fourier-Net [21]	0.763±0.129	**2.857±0.456**	**0.748±0.114**	0.024±0.019	0.672±0.022	6.716±0.601	1.666±0.180	0.216±0.104	4.20M
LapIRN [31]	0.763±0.133	3.166±0.608	0.779±0.126	0.312±0.106	0.716±0.016	6.116±0.454	1.426±0.133	0.024±0.009	1.20M
PRNet++ [23]	0.755±0.130	3.593±0.748	0.857±0.162	1.052±0.302	0.701±0.021	6.492±0.597	1.520±0.177	0.072±0.027	1.24M
ModeT [46]	0.758±0.125	3.496±0.732	0.828±0.151	0.114±0.057	0.721±0.013	5.969±0.416	1.375±0.110	0.010±0.004	1.03M
Im2Grid [25]	0.761±0.127	3.316±0.668	0.799±0.128	<0.0002	0.713±0.014	6.062±0.428	1.419±0.118	0.007±0.003	0.89M
RDN [18]	0.759±0.123	3.476±0.802	0.823±0.161	<0.0001	0.713±0.017	6.208±0.497	1.436±0.142	<0.0002	28.65M
Ours	**0.769±0.127**	3.128±0.631	0.760±0.125	0.083±0.045	**0.731±0.012**	**5.862±0.436**	**1.326±0.114**	0.018±0.006	1.11M

Novel ways to Improve SACB-Net

- SACB-Net rely on k-means clustering. It can miss long range dependencies. So, we can reply it with State Space Models like Mamba which provides a global receptive field. (Reference)

- SACB-Net learns its features from scratch. Hence it can be affected by noise. So, instead we can use a Segment Anything Model (SAM). SAM is pre-trained on images and is more robust. (Reference)

Thank You!