Sparsity in CNNs

Journal Club (11th Dec 2017)

Marginal Space Deep Learning: Efficient Architecture for Volumetric Image Parsing

9-Parameter Affine Transform

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

4x4 translation matrix **T** (3 parameters)

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

4x4 scale matrix **S** (3 parameters)

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\phi_x) & -\sin(\phi_x) & 0 \\ 0 & \sin(\phi_x) & \cos(\phi_x) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

rotation matrix **Rx** for rotation about x-axis (1 parameter)

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\phi_y) & 0 & -\sin(\phi_y) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\phi_y) & 0 & \cos(\phi_y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

rotation matrix **Ry** for rotation about y-axis (1 parameter)

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\phi_z) & -\sin(\phi_z) & 0 & 0 \\ \sin(\phi_z) & \cos(\phi_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

rotation matrix **Rz** for rotation about z-axis (1 parameter)

M= [S][Rx][Ry][Rz][T]

Above ordering does translations first to match origins, then rotations about new origin, and finally scaling of the aligned image. Most software used 9 or 12 parameters (sometimes called degrees of freedom) for registration.

Problem

Segmenting aortic valve in TEE images

Volumes vary from 100x100x50 to 250x250x150 voxels, and 0.75 to 1mm

2891 volumes from 869 patients

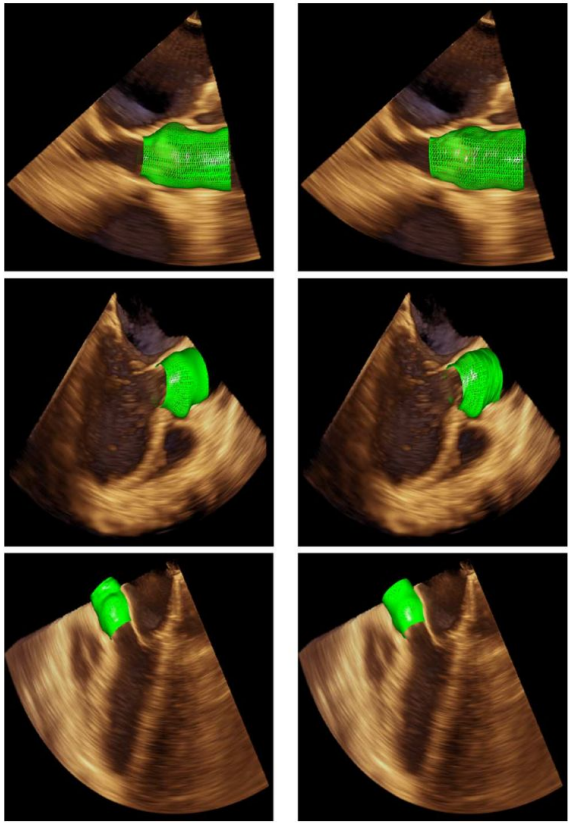


Fig. 9. Example images showing the aortic valve segmentation results for different patients from the test set, using our proposed method. Each row of images corresponds to a different patient, the left image represents the groundtruth mesh, while the right image shows the detected mesh.

Estimate 9D space

Marginal Space Deep Learning

Learning classifiers in clustered, high probability regions of spaces

1) Roughly initialize the 9 parameters (localization)
2) Used deep learning to “guide the shape deformation” (fine-tuning)

Sparse Adaptive Deep Neural Network (SADNN)

Extracting sufficiently representative training set

Training set = m patches with a set of class assignment (y)

Find a sparsity map for the weights (w)

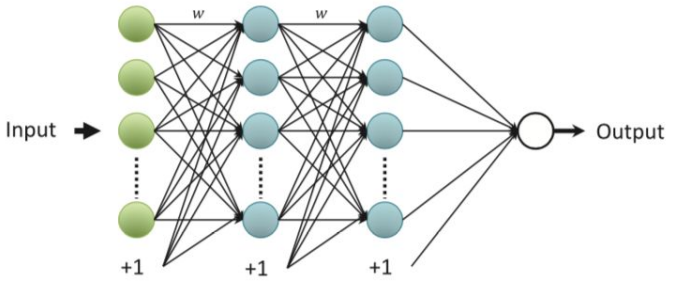


Fig. 2. Example of a fully connected neural network with 3 layers. Every neuron in a layer is connected to all neurons in the previous layer.

Algorithm 1 Learning algorithm with iterative threshold-enforced sparsity

Pre-training using all weights $\vec{w}^{(0)} \leftarrow \vec{w}$ (small # epochs)
Initialize sparsity map $\vec{s}^{(0)}$ with ones
$t \leftarrow 1$
for training round $t \leq T$ **do**
    for all filters i with sparsity **do**
        $\vec{s}_i^{(t)} \leftarrow \vec{s}_i^{(t-1)}$ + remove smallest active weights
        $\vec{w}_i^{(t)} = \vec{w}_i^{(t-1)} \odot \vec{s}_i^{(t)}$
        Normalize active coeff. s.t. $\|\vec{w}_i^{(t)}\|_1 = \|\vec{w}_i^{(t-1)}\|_1$
    end for
    $\vec{b}^{(t)} \leftarrow \vec{b}^{(t-1)}$ (copy current biases)
    Train network on active weights (small # epochs)
    $t \leftarrow t + 1$
end for
Sparse kernels: $\vec{w}_s \leftarrow \vec{w}^{(T)}$
Biases: $\vec{b}_s \leftarrow \vec{b}^{(T)}$

Algorithm 1 Learning algorithm with iterative threshold-enforced sparsity

Pre-training using all weights $\vec{w}^{(0)} \leftarrow \vec{w}$ (small # epochs)
Initialize sparsity map $\vec{s}^{(0)}$ with ones
$t \leftarrow 1$
for training round $t \leq T$ **do**
    for all filters i with sparsity **do**
        $\vec{s}_i^{(t)} \leftarrow \vec{s}_i^{(t-1)}$ + remove smallest active weights
        $\vec{w}_i^{(t)} = \vec{w}_i^{(t-1)} \odot \vec{s}_i^{(t)}$
        Normalize active coeff. s.t. $\|\vec{w}_i^{(t)}\|_1 = \|\vec{w}_i^{(t-1)}\|_1$
    end for
    $\vec{b}^{(t)} \leftarrow \vec{b}^{(t-1)}$ (copy current biases)
    Train network on active weights (small # epochs)
    $t \leftarrow t + 1$
end for
Sparse kernels: $\vec{w}_s \leftarrow \vec{w}^{(T)}$
Biases: $\vec{b}_s \leftarrow \vec{b}^{(T)}$

Algorithm 1 Learning algorithm with iterative threshold-enforced sparsity

Pre-training using all weights $\vec{w}^{(0)} \leftarrow \vec{w}$ (small # epochs)
Initialize sparsity map $\vec{s}^{(0)}$ with ones
$t \leftarrow 1$
for training round $t < T$ **do**
    for all filters i with sparsity **do**
        $\vec{s}_i^{(t)} \leftarrow \vec{s}_i^{(t-1)}$ + remove smallest active weights
        $\vec{w}_i^{(t)} = \vec{w}_i^{(t-1)} \odot \vec{s}_i^{(t)}$
        Normalize active coeff. s.t. $\|\vec{w}_i^{(t)}\|_1 = \|\vec{w}_i^{(t-1)}\|_1$
    end for
    $\vec{b}^{(t)} \leftarrow \vec{b}^{(t-1)}$ (copy current biases)
    Train network on active weights (small # epochs)
    $t \leftarrow t + 1$
end for
Sparse kernels: $\vec{w}_s \leftarrow \vec{w}^{(T)}$
Biases: $\vec{b}_s \leftarrow \vec{b}^{(T)}$

"...in later stages of the training exponentially less filter weights are set to zero "

Dropout: Similar end result

Translation Invariance: No convs. BUT each sparse pattern in invariant to position so we get translation invariance

Pooling: None here

Kernel structure: No longer square because we have an adaptive sampling method

Marginal Space Deep Learning

$$\left(\hat{\vec{T}}, \hat{\vec{R}}, \hat{\vec{S}}\right) = \arg\max_{\vec{T},\vec{R},\vec{S}} p(\vec{T}|I)p(\vec{R}|\vec{T},I)p(\vec{S}|\vec{T},\vec{R},I)$$

$$= \arg\max_{\vec{T},\vec{R},\vec{S}} p(\vec{T}|I)\frac{p(\vec{T},\vec{R}|I)}{p(\vec{T}|I)}\frac{p(\vec{T},\vec{R},\vec{S}|I)}{p(\vec{T},\vec{R}|I)},$$

Marginal spaces

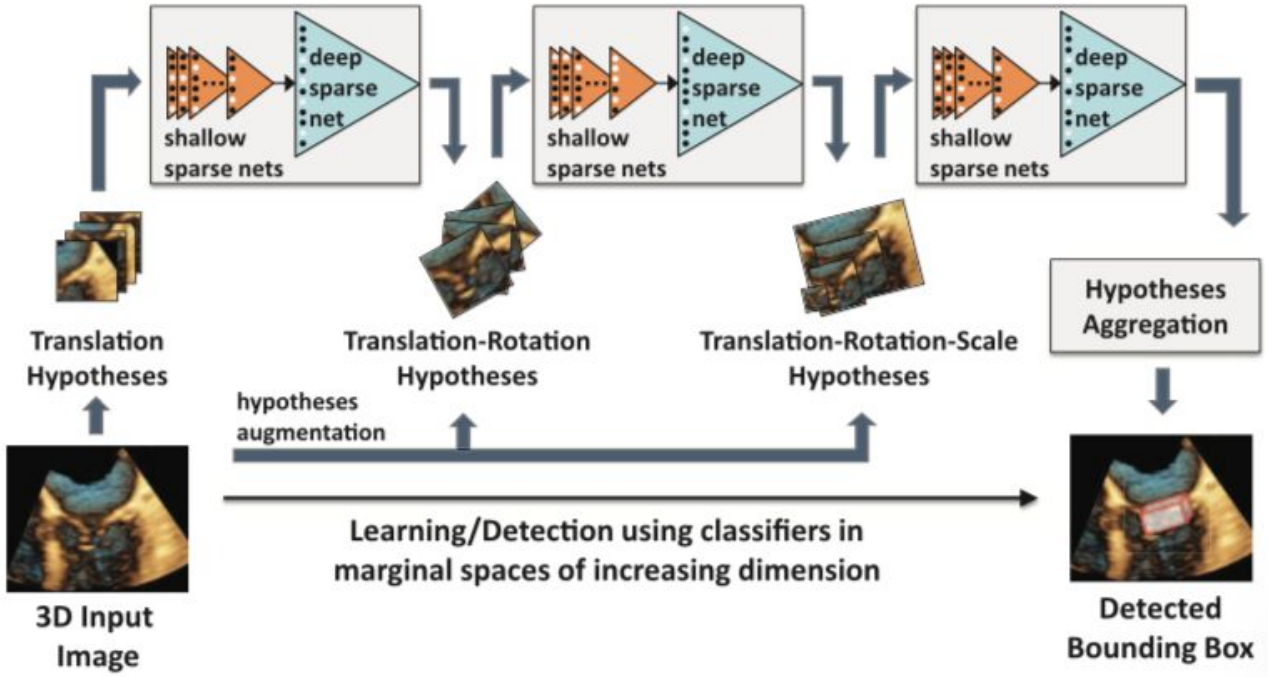


Fig. 4. Schematic visualization of the marginal space deep learning framework. The black/white dots encode the sparse sampling patterns.

Balancing training set

Due to limited range in position, orientation and space, great class imbalance exists (1:1000)

Series of shallow nets

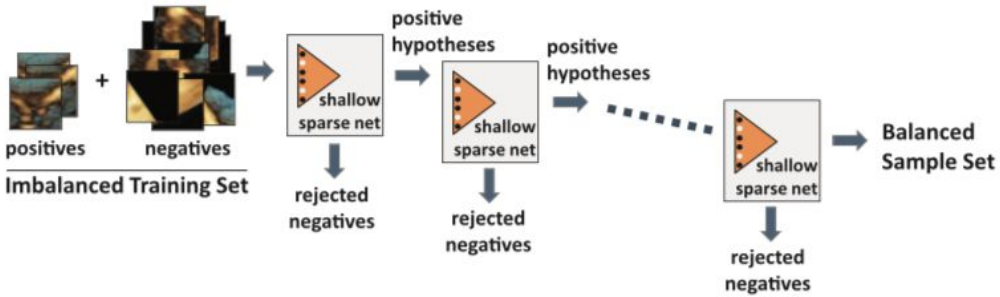


Fig. 5. Schematic visualization of the negative-sample filtering cascade used to balance the training set. The black/white dots encode the sparse sampling patterns of the shallow nets.

Boundary estimation

Boundary classification problem using translation and orientation along the normal

Training data: samples along the ground truth boundary (positive) and various distances from boundary (negative)

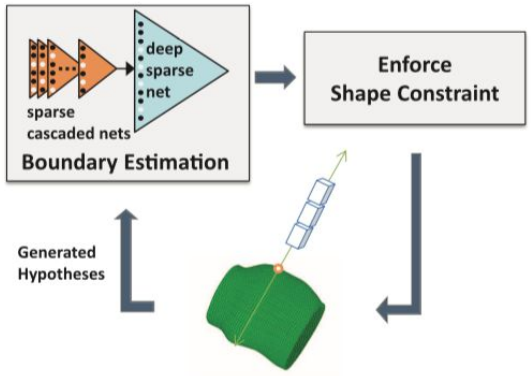
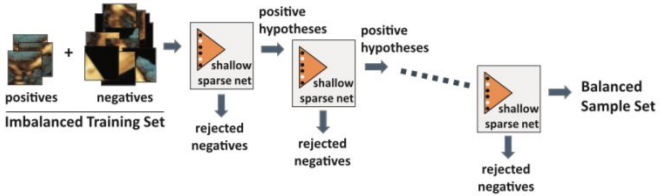


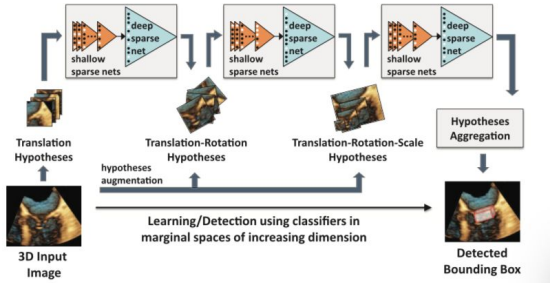
Fig. 6. Schematic visualization of the boundary deformation with SADNN. Starting from the current shape, the SADNN is aligned and applied along the normal for each point of the mesh, the boundary is deformed and projected under the current shape space. The process is iteratively repeated. The black/white dots encode the sparse patterns, learned in the cascaded shallow networks and deep boundary classifier.

Experiments

Cascade network: 5832 (sparse) x 60 x 1



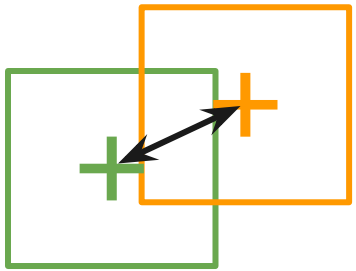
Main classifier: 5832 (sparse) x 150 x 80 x 50 x 1



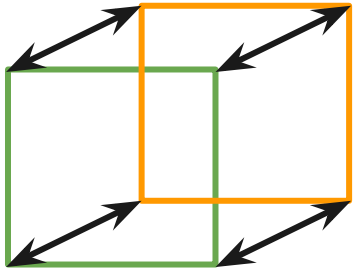
Experiments

Cascade network: 5832 (sparse) x 60 x 1

Main classifier: 5832 (sparse) x 150 x 80 x 50 x 1



translation



Orientation and scale

TABLE I

COMPARISON OF THE PERFORMANCE OF THE STATE-OF-THE-ART MSL [5] AND THE PROPOSED MSDL FRAMEWORK FOR AORTIC VALVE DETECTION. THE MEASURES USED TO QUANTIFY THE QUALITY OF THE RESULTS W.R.T TO THE GROUND TRUTH DATA ARE THE ERROR OF THE POSITION OF THE BOX AND MEAN CORNER DISTANCE (BOTH MEASURED IN MILLIMETRES). THE SUPERIOR RESULTS ARE DISPLAYED IN BOLD

	Position Error [mm]				**Corner Error [mm]**			
	Training Data		**Test Data**		**Training Data**		**Test Data**	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	3.12	**1.47**	3.34	**1.83**	5.42	**2.80**	6.16	**3.72**
Median	2.80	**1.27**	3.05	**1.58**	4.98	**2.58**	5.85	**3.34**
STD	1.91	**0.99**	1.85	**1.31**	2.47	**1.23**	2.31	**1.74**

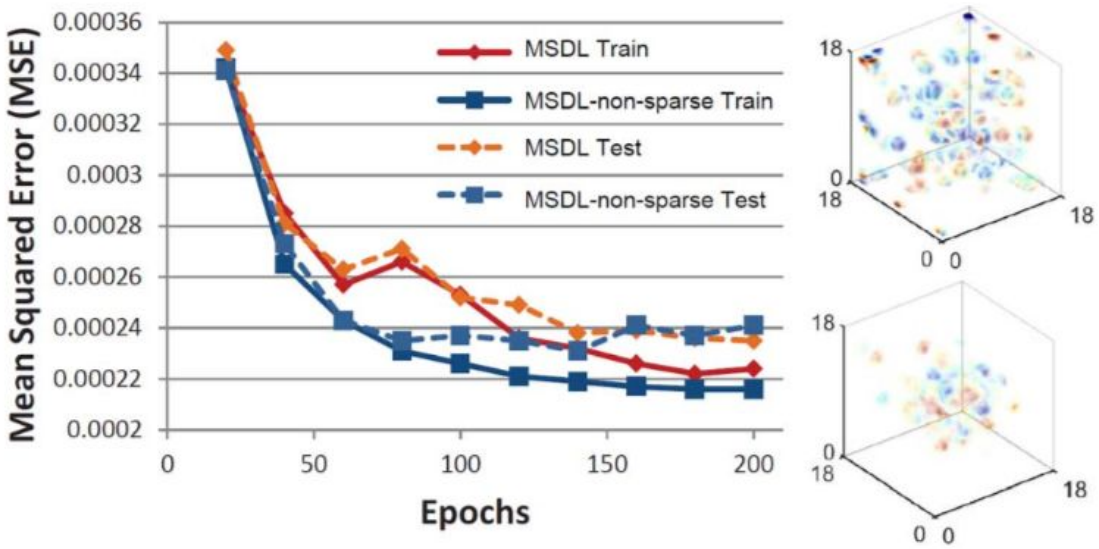


Fig. 7. Left: Training and testing error of the SADNN in the translation stage. The performance oscillation of the sparse model is explained by the enforcement of sparsity in the dense sampling layer of the network. Right: Example color-coded sparse patterns for translation (upper box) and full space (lower box). The latter representation is more compact because of the better data alignment.

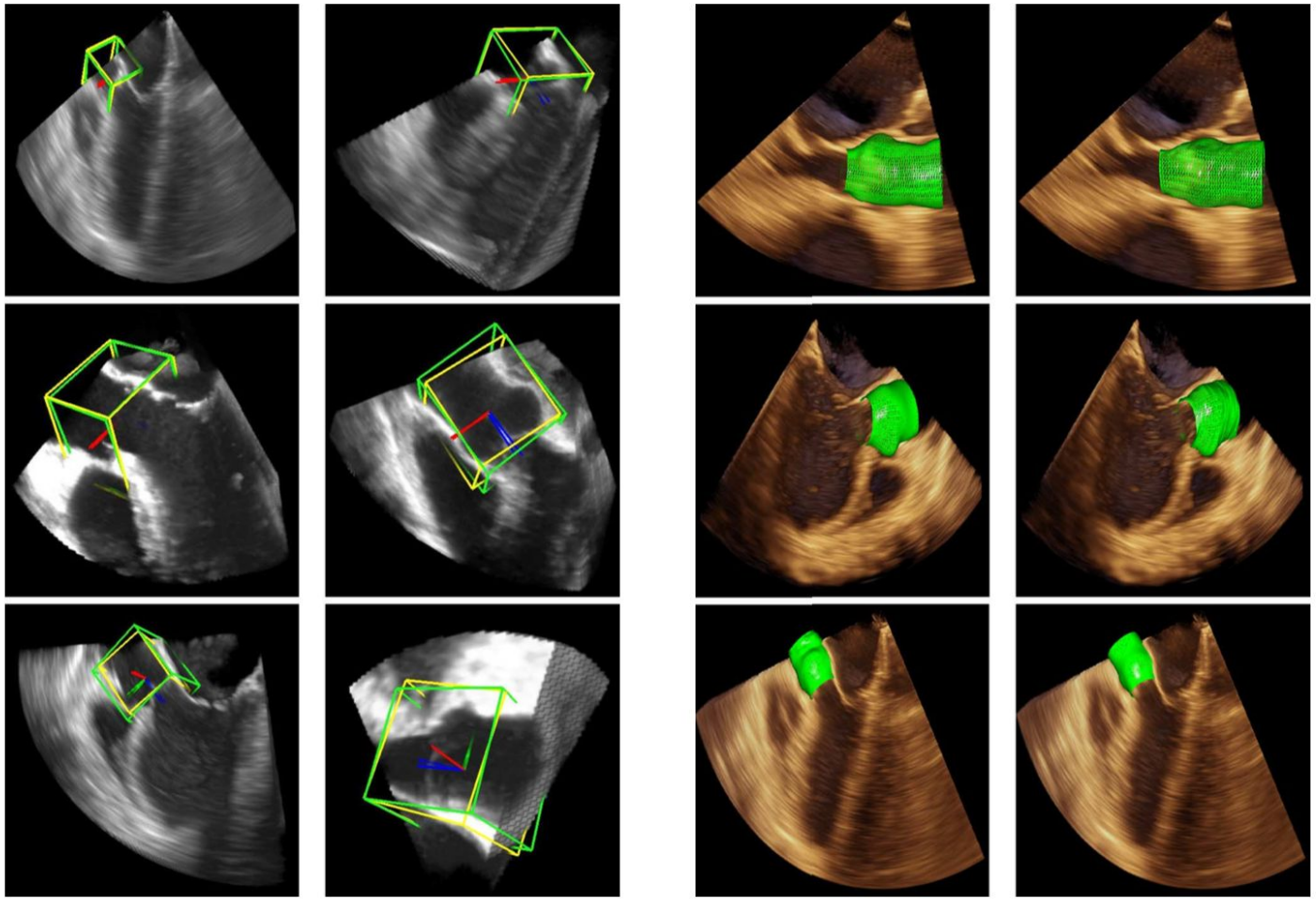


TABLE II

COMPARISON OF THE PERFORMANCE OF THE STATE-OF-THE-ART MSL [5] AND THE PROPOSED MSDL FRAMEWORK FOR AORTIC VALVE SEGMENTATION. THE MEASURE ILLUSTRATES THE DISTANCE OF THE DETECTED MESH TO THE GROUNDTRUTH FOR BOTH THE INITIALIZATION FROM THE DETECTED BOX AS WELL AS THE DISTANCE AFTER BOUNDARY REFINEMENT. THE SUPERIOR RESULTS ARE DISPLAYED IN BOLD

	Training Data, distance to ground truth mesh [mm]				**Test Data, distance to ground truth mesh [mm]**			
	Initialization		**Final**		**Initialization**		**Final**	
	MSL	MSDL	MSL	MSDL	MSL	MSDL	MSL	MSDL
Mean	2.08	**1.16**	1.17	**0.89**	2.06	**1.21**	1.04	**0.90**
Median	1.94	**1.09**	1.05	**0.82**	1.95	**1.10**	0.98	**0.80**
STD	0.83	**0.40**	0.66	**0.35**	0.79	**0.55**	0.50	**0.48**

TABLE III

COMPARISON OF THE PERFORMANCE OF OUR DL-BASED FRAMEWORK WITH THE STATE-OF-THE-ART SOLUTION [5] AND THE TWO MIXED VARIANTS: COMBINING THE MSDL BOX DETECTION WITH THE ORIGINAL SEGMENTATION METHOD [5], RESPECTIVELY THE MSL BOX DETECTION [5] WITH OUR DL-BASED SEGMENTATION METHOD. THE SUPERIOR RESULTS ARE DISPLAYED IN BOLD

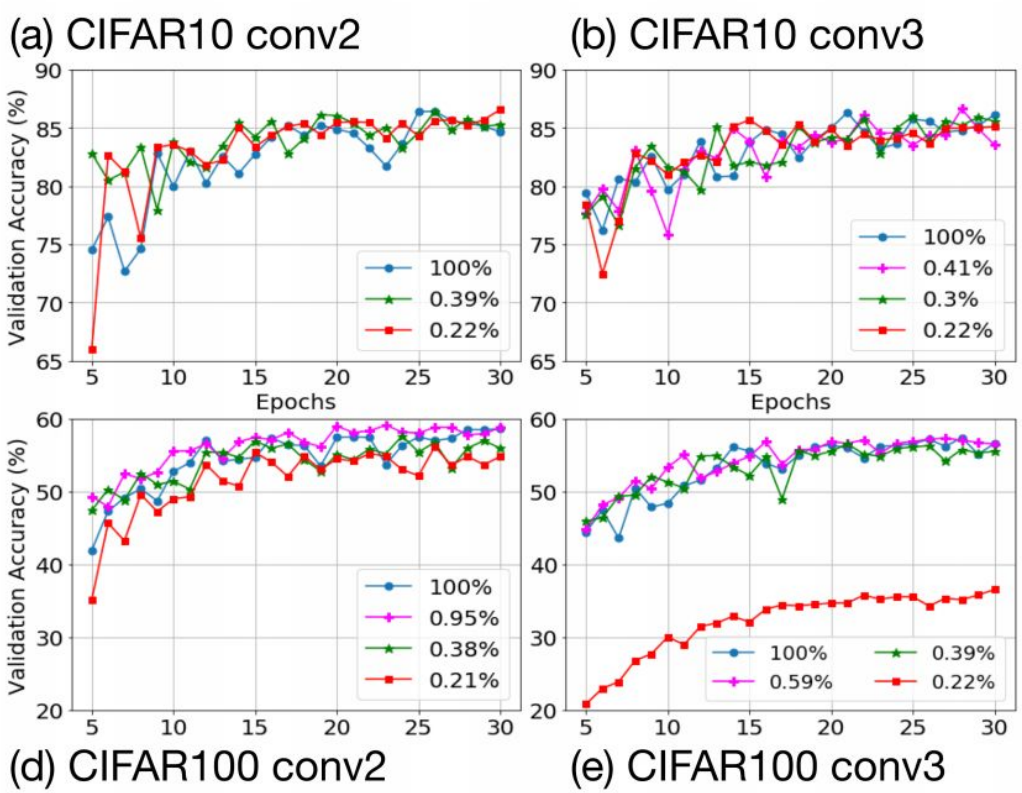
	Segmentation error [mm]		
	Mean	Median	STD
Ours	**0.90**	**0.80**	**0.48**
Reference [5]	1.04	0.98	0.50
Detection (ours) + Seg. [5]	0.95	0.88	0.48
Detection [5] + Seg. (ours)	1.00	0.90	0.51

Characterizing Sparse Connectivity Patterns in Neural Networks

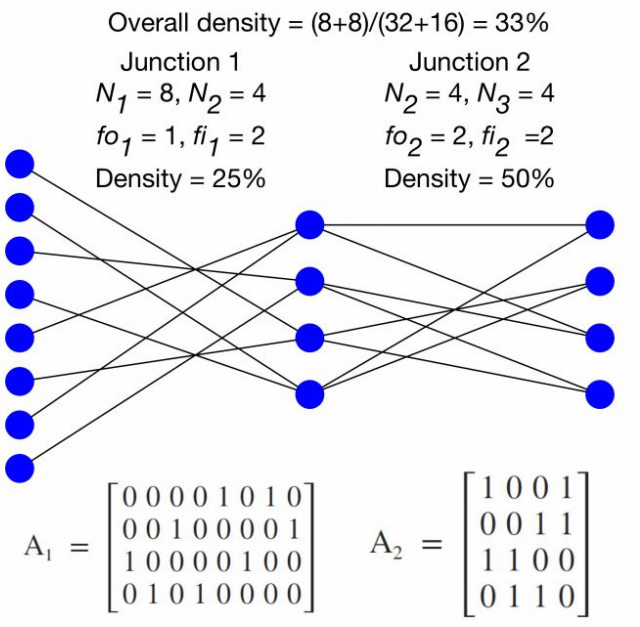
https://arxiv.org/abs/1711.02131

Metric for measuring sparsity in fully connected layers in CNN

If you reduced connectivity in fully connected layers, it made little difference to overall performance



Window Adjacent Matrix (A^w)



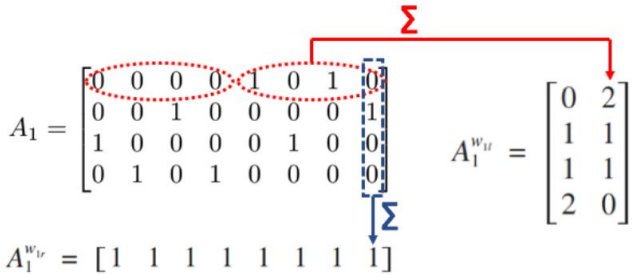


Figure 6: Construction of window adjacency matrices.

Scatter (S)

= Proportion of values > 1 in window adjacent matrix

We need to consider the number of S values in the network i.e. the number of junction which share the same level of scatter

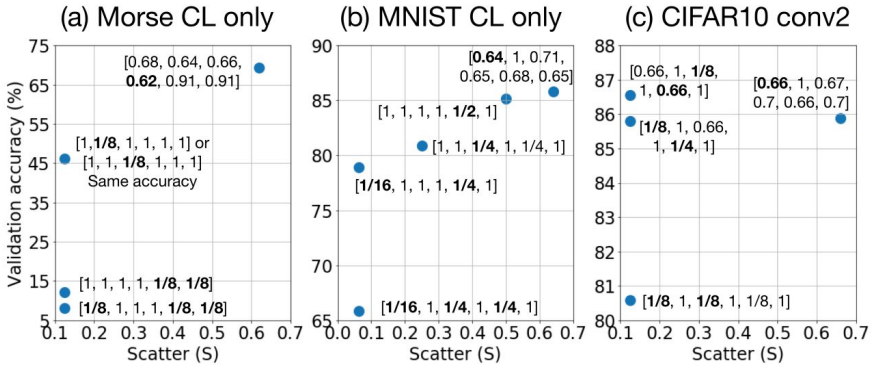


Figure 7: Network performance vs. scatter for CL only networks of (a) Morse (b) MNIST, and convolutional network with 2 CL junctions of (c) CIFAR10. All minimum values that need to be considered to differentiate between connection patterns are bolded.

Future Work

Extending this concept to conv layers

In the meantime, small Keras script will be made available on martellab website