Locating Cephalometric X-Ray Landmarks with Foveated Pyramid Attention

Logan Gilmour, Nilanjan Ray
University of Alberta

MIDL 2020

The problem we're solving:

One of the existing best methods [1] uses 2 different scales of Random Forest regression using Haar features.

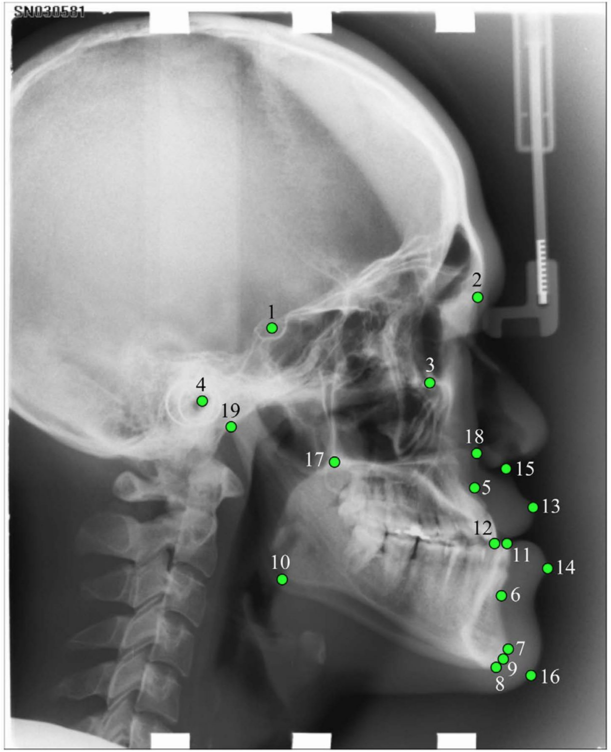
Another best method uses 2 scales of U-Net.

Suggests a multiresolution approach might work well.

Images are 2400 x 1935.

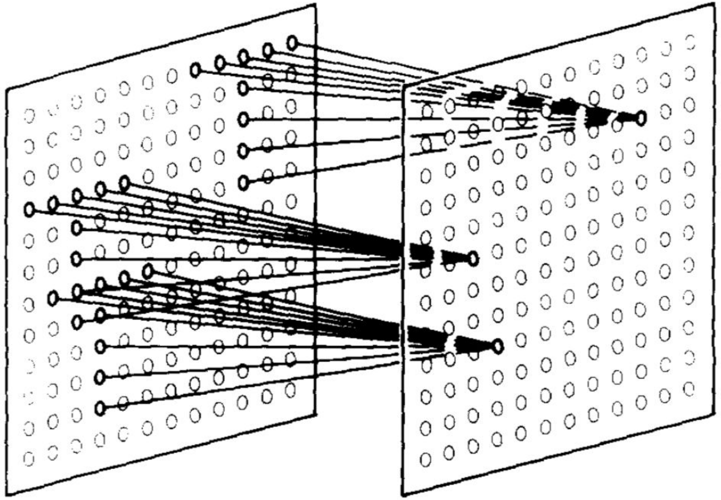
[1]C. Lindner, C.-W. Wang, C.-T. Huang, C.-H. Li, S.-W. Chang, and T. F. Cootes, "Fully Automatic System for Accurate Localisation and Analysis of Cephalometric Landmarks in Lateral Cephalograms," *Scientific Reports*, vol. 6, no. 1, Sep. 2016.

[2] Z. Zhong, J. Li, Z. Zhang, Z. Jiao, and X. Gao, "An Attention-Guided Deep Regression Model for Landmark Detection in Cephalograms," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, vol. 11769, D. Shen, T. Liu, T. M. Peters, L. H. Staib, C. Essert, S. Zhou, P.-T. Yap, and A. Khan, Eds. Cham: Springer International Publishing, 2019, pp. 540–548.

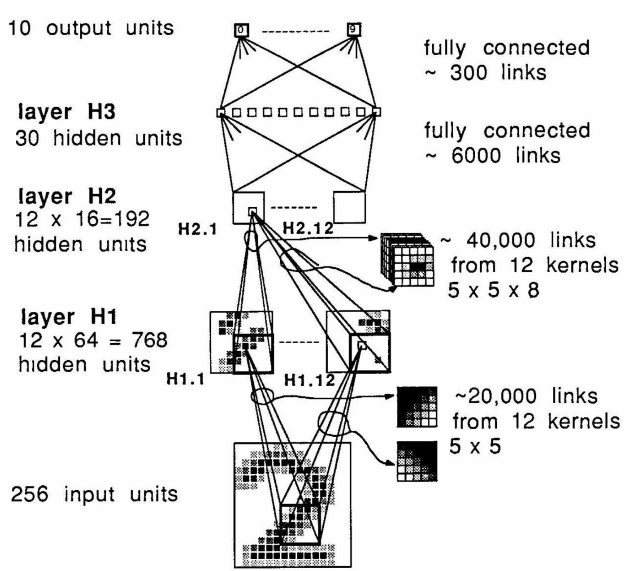


L1	Sella
L2	Nasion
L3	Orbitale
L4	Porion
L5	Subspinale
L6	Supramentale
L7	Pogonion
L8	Menton
L9	Gnathion
L10	Gonion
L11	Incision inferius
L12	Incision superius
L13	Upper lip
L14	Lower lip
L15	Subnasale
L16	Soft tissue pogonion
L17	Posterior nasal spine
L18	Anterior nasal spine
L19	Articulare

CNNs were originally inspired by human vision.



Neocognitron [1]

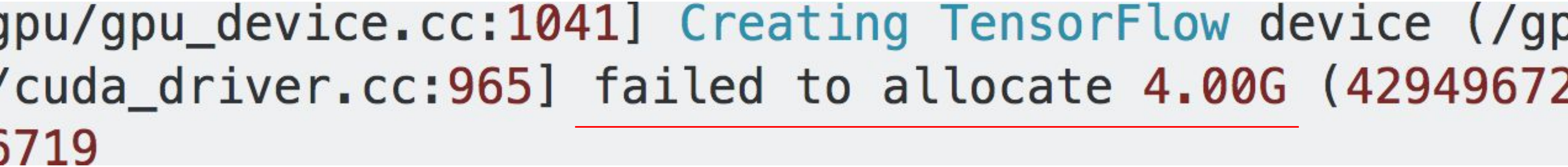


Backprop in a CNN [2]

[1] K. Fukushima, “Neocognitron: A hierarchical neural network capable of visual pattern recognition,” *Neural Networks*, vol. 1, no. 2, pp. 119–130, Jan. 1988.

[2] Y. LeCun *et al.*, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

But for big images...



```
gpu/gpu_device.cc:1041] Creating TensorFlow device (/gp
/cuda_driver.cc:965] failed to allocate 4.00G (42949672
5719
```

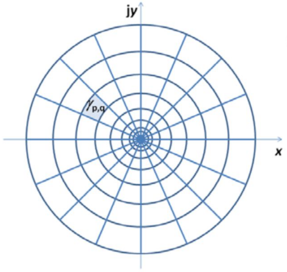
Even recently, “big” is 480 x 480 [1]

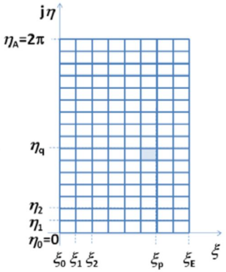
If we are interested in regression problems in high resolution images, this isn’t great.

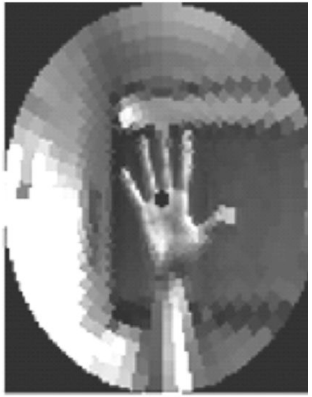
[1] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” *arXiv:1905.11946 [cs, stat]*, Nov. 2019.

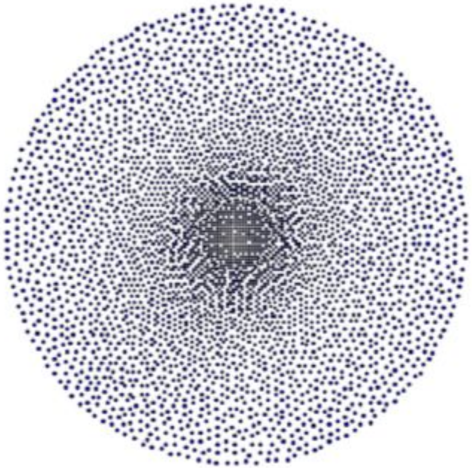
Still a key difference: Uniform Sampling

Mammalian vision has been shown to have roughly log-polar sampling density, centered on the fovea:









Left 3: V. Javier Traver and A. Bernardino, “A review of log-polar imaging for visual perception in robotics,” *Robotics and Autonomous Systems*, vol. 58, no. 4, pp. 378–398, Apr. 2010.

Right 2: P. Ozimek, L. Balog, R. Wong, T. Esparon, and J. P. Siebert, “Egocentric Perception using a Biologically Inspired Software Retina Integrated with a Deep CNN,” in *International Conference on Computer Vision 2017, ICCV 2017, Second International Workshop on Egocentric Perception, Interaction and Computing*, 2017.

Problem

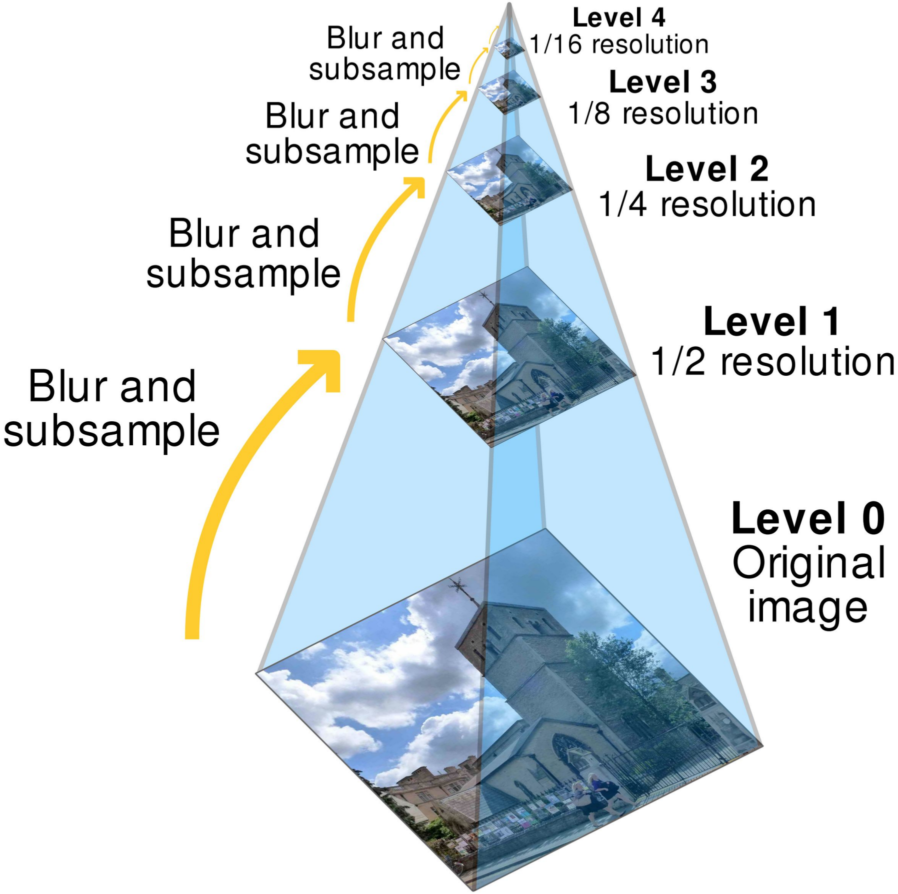
No longer translation invariant. Not necessarily a huge problem except…

Transfer learning significantly less effective!

Another Approach:

Image Pyramids

Give us a representation with both coarse and fine detail



https://en.wikipedia.org/wiki/Pyramid_%28image_processing%29#/media/File:Image_pyramid.svg

Wait!

That’s more pixels, not less!

Because of the memory costs, existing approaches that use pyramids typically use them only at inference time, or attempt to construct them incidentally along with features. [1]

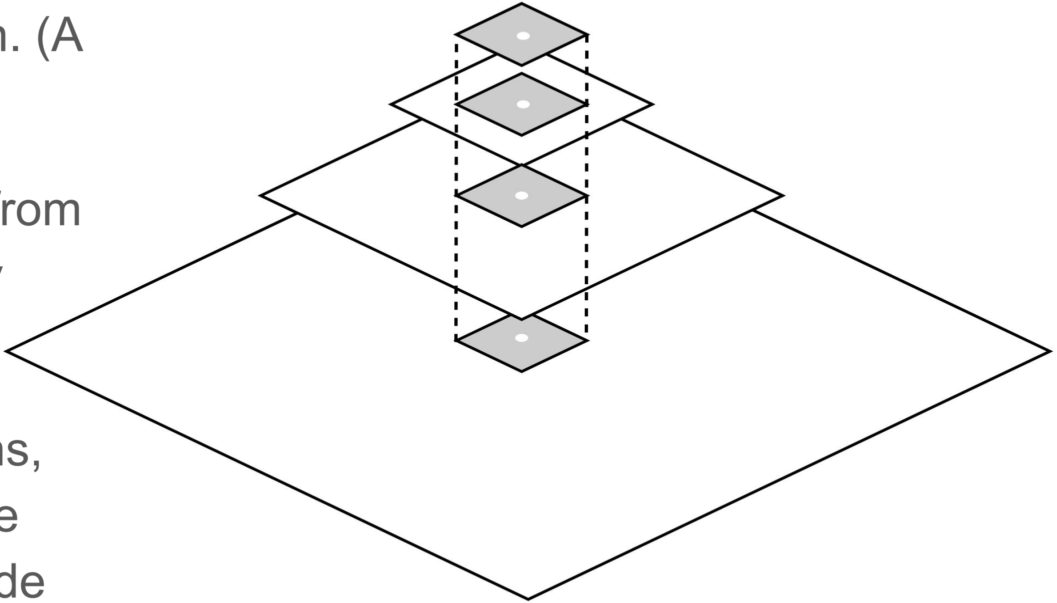
[1] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature Pyramid Networks for Object Detection,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 936–944.

We’ll throw most of them away!

Take a 64 x 64 patch from each, centered on the same location. (A glimpse)

If we predict incorrectly, start from new predicted position and try again.

For a fixed number of iterations, problem scales with log of side length, instead of square of side length!



Proposed Method:

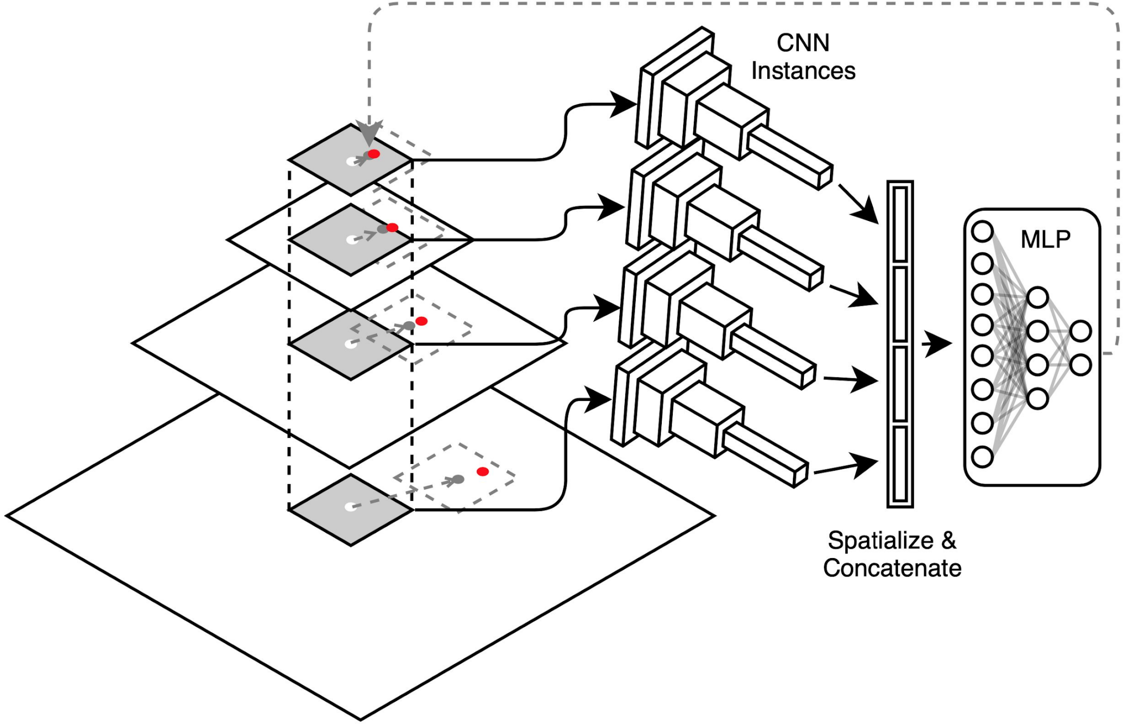
Trying to regress to target red dot:

1. Make a Gaussian Pyramid from input Image

2. CNNs get image patches centered on an initial estimate of landmark location (initialized at center of image)

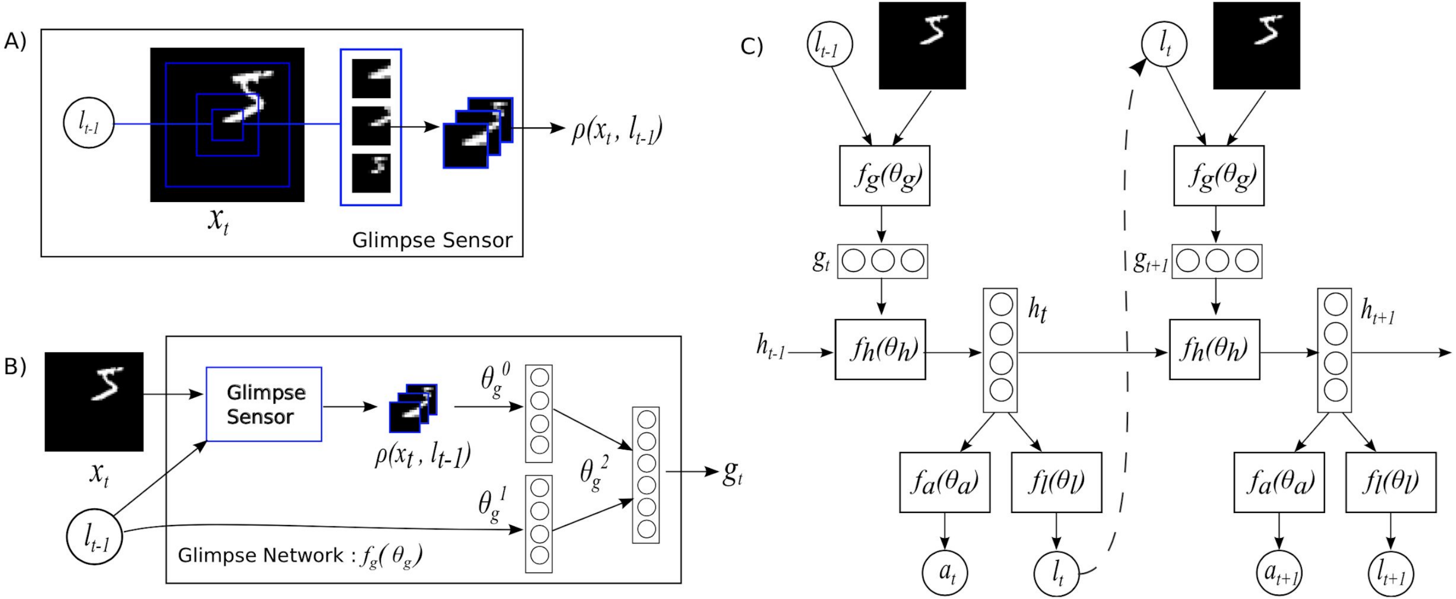
3. They produce features used to predict an offset from their current location (grey dot)

4. Repeat from step 2 using new location (estimate + predicted error)



Related Work

Will it work? Existing work: Recurrent Models of Visual Attention [1]



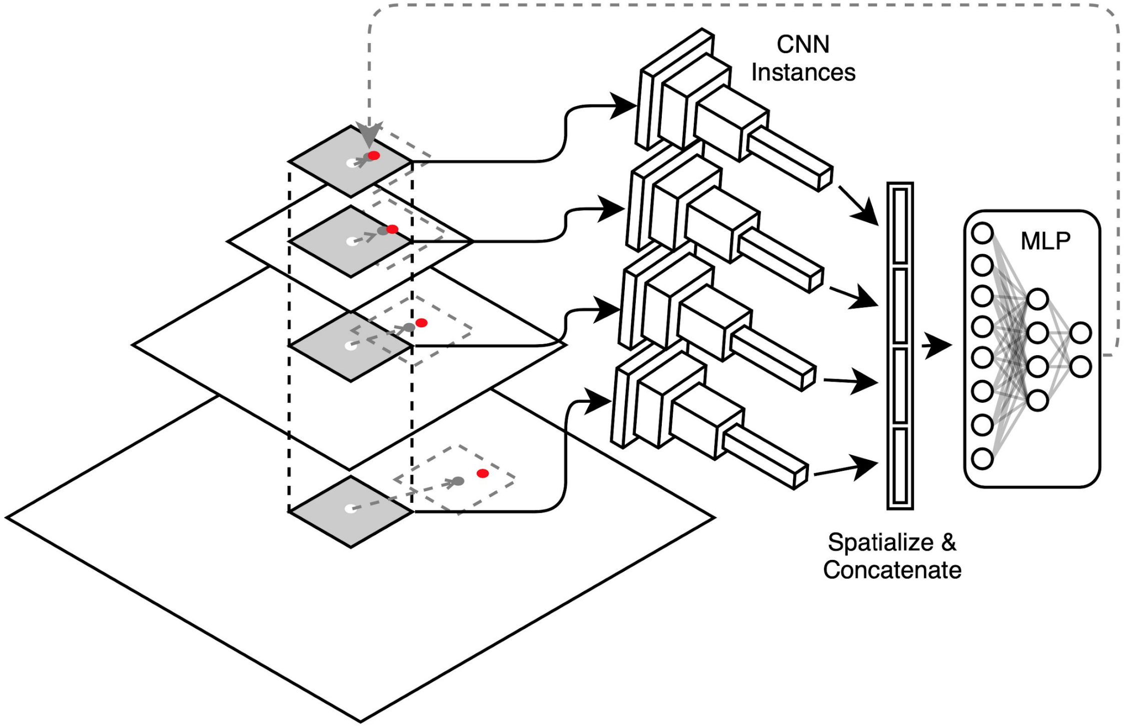
[1] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, “Recurrent Models of Visual Attention,” *arXiv:1406.6247 [cs, stat]*, Jun. 2014.

Pyramid

Gaussian Pyramid is downsampled by a factor of 2 at each level.

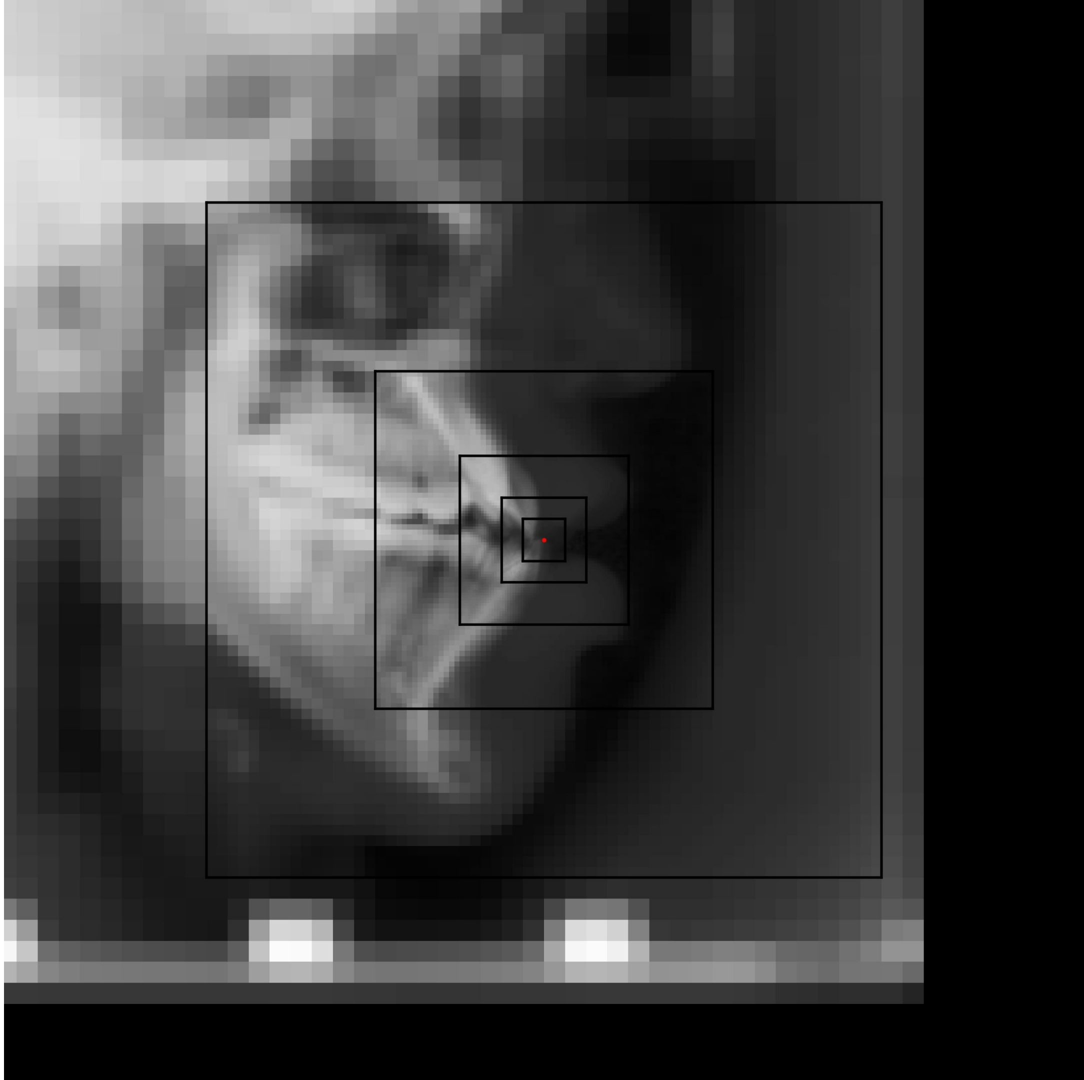
Patches in the glimpse (grey) are 64 x 64.

There are enough levels that the top of the pyramid roughly fits in a 64 x 64 glimpse.



Visualization

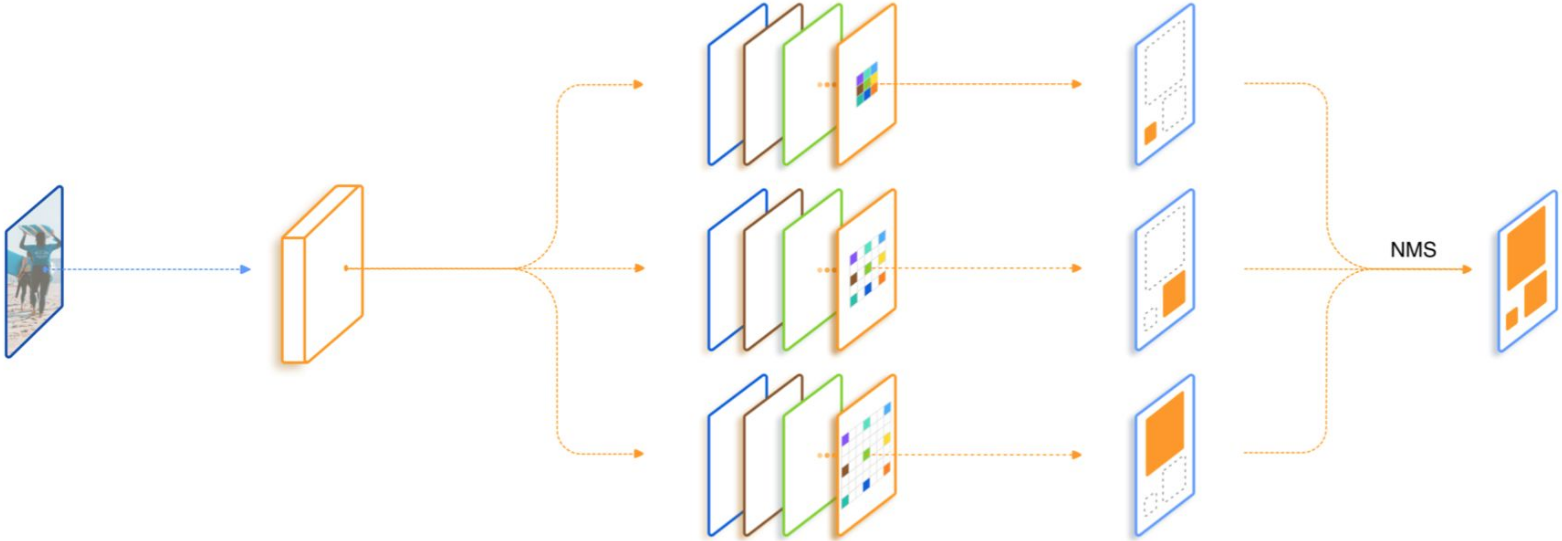
What the network ‘sees’ when centered on the red dot (a landmark for the bottom incisor)



Related Work

We want to use a CNN. What should it look like?

We use an idea from Trident Networks (specifically weight sharing).



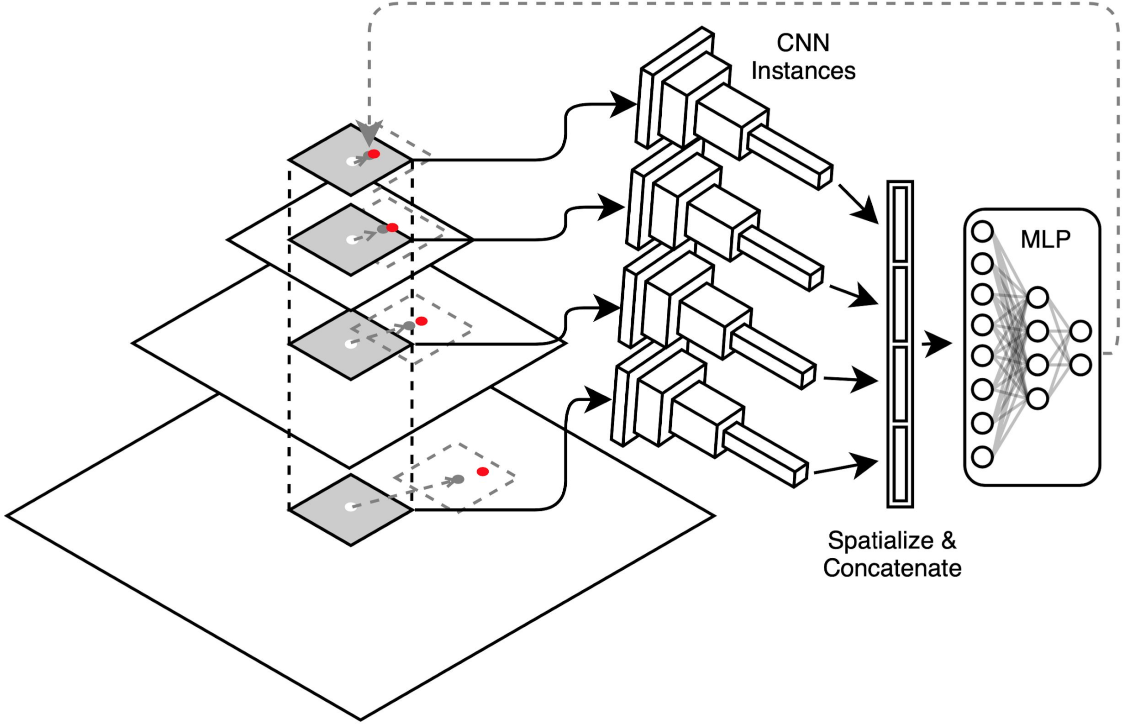
Y. Li, Y. Chen, N. Wang, and Z. Zhang, “Scale-Aware Trident Networks for Object Detection,” *arXiv:1901.01892 [cs]*, Aug. 2019.

CNN

CNNs are ResNet-34 with final three Basic Blocks and fully connected layer removed. This removes 2 downsamples.

Stride of input layer is reduced from 2 to 1. This effectively removes another downsample.

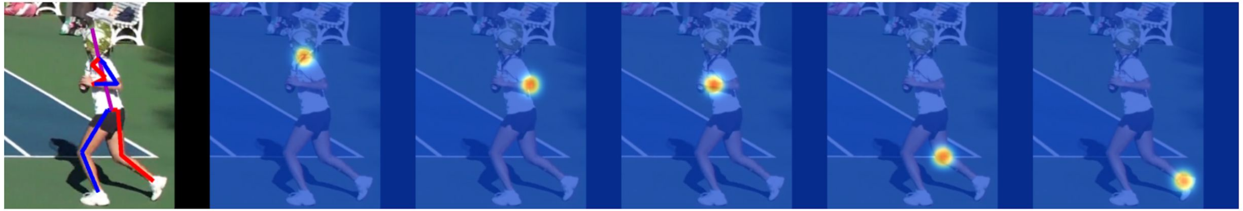
For a 64 x 64 patch input, the resulting activation volume is 256 x 8 x 8.



Related Work

What does modern CNN regression look like?

Heatmap Regression for Pose detection [1]:



Reformulating heatmap max as expectation [2]:

$$\mathbf{J}_k = \sum_{p_z=1}^{D} \sum_{p_y=1}^{H} \sum_{p_x=1}^{W} \mathbf{p} \cdot \tilde{\mathbf{H}}_k(\mathbf{p}),$$

$$\tilde{\mathbf{H}}_k(\mathbf{p}) = \frac{e^{\mathbf{H}_k(\mathbf{p})}}{\int_{\mathbf{q} \in \Omega} e^{\mathbf{H}_k(\mathbf{q})}}.$$

[1] A. Newell, K. Yang, and J. Deng, “Stacked Hourglass Networks for Human Pose Estimation,” *arXiv:1603.06937 [cs]*, Jul. 2016.

[2] X. Sun, B. Xiao, F. Wei, S. Liang, and Y. Wei, “Integral Human Pose Regression,” in *Computer Vision – ECCV 2018*, vol. 11210, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 536–553.

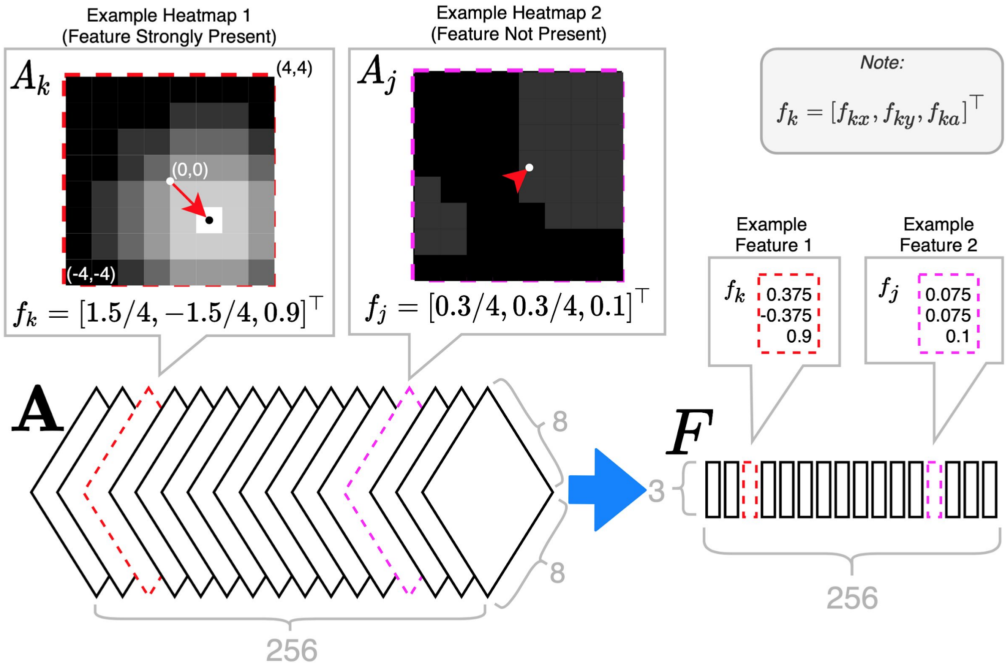
Spatialized Features

Treat each 8x8 activation as a probability distribution (via softmax), and find the expected value of its x,y coordinates (Center of Mass).

Additionally, find the expected value of the raw activations to determine overall feature intensity, as maybe it's not actually present in the patch. (A 'soft-max-pool').

$$\boldsymbol{f}_i = \sum_{y=1}^{H=8} \sum_{x=1}^{W=8} p_i(x,y) \begin{bmatrix} x \\ y \\ A_i(x,y) \end{bmatrix}$$

Output is reduced to 3 x 256.

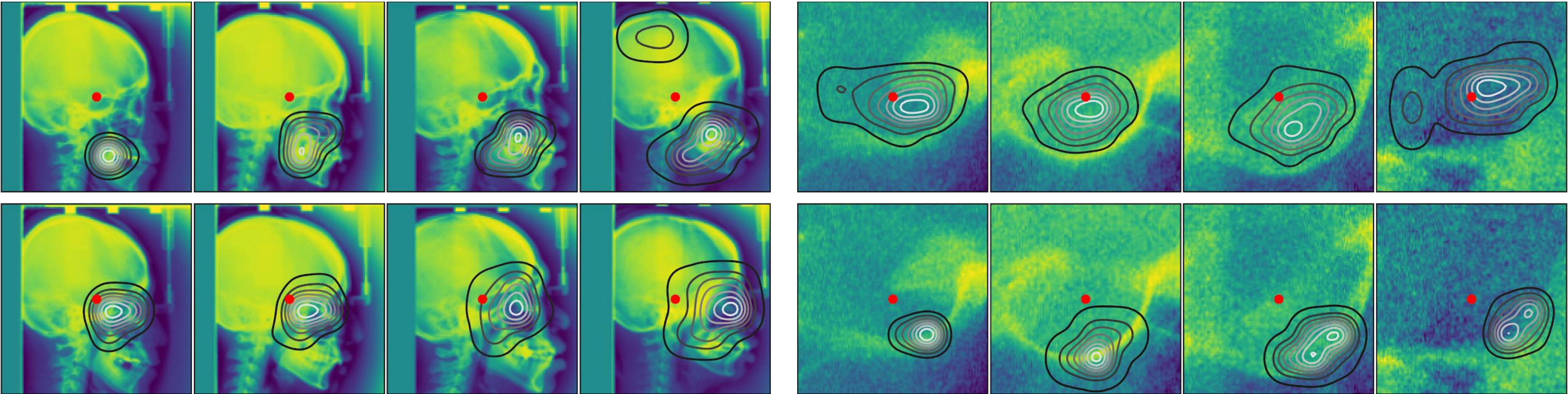


Spatialized Features

Some visualizations of the heatmaps learned by integral regression.

Each quadrant is a different feature (with four example 2D activation maps).

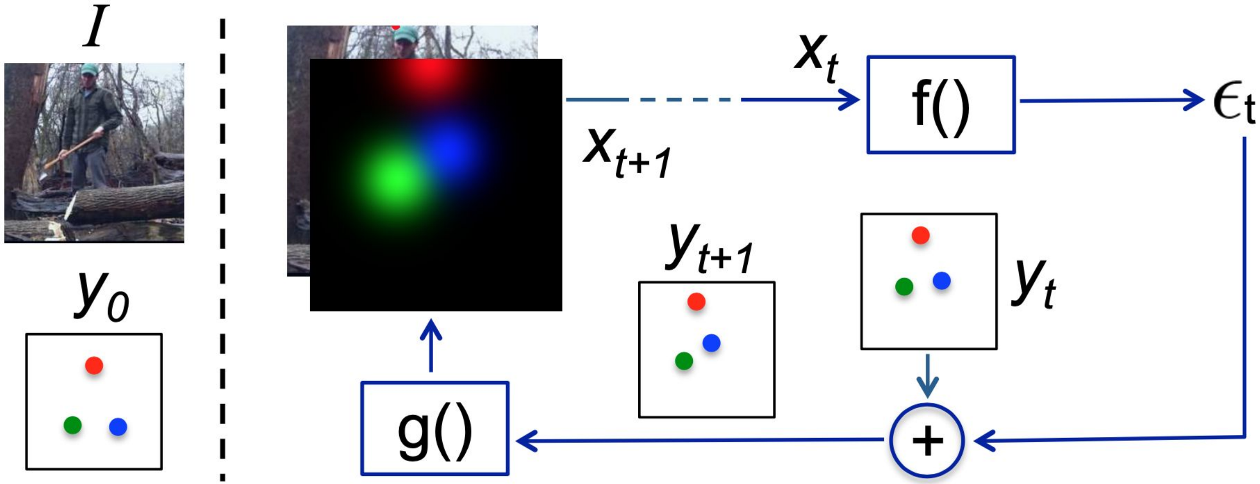
Red dot is ground truth.



Related Work

How do we chose where to look?

Iterative Error Feedback for Human Pose Regression [1]



[1] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik, “Human Pose Estimation with Iterative Error Feedback,” *arXiv:1507.06550 [cs]*, Jun. 2016.

MLP

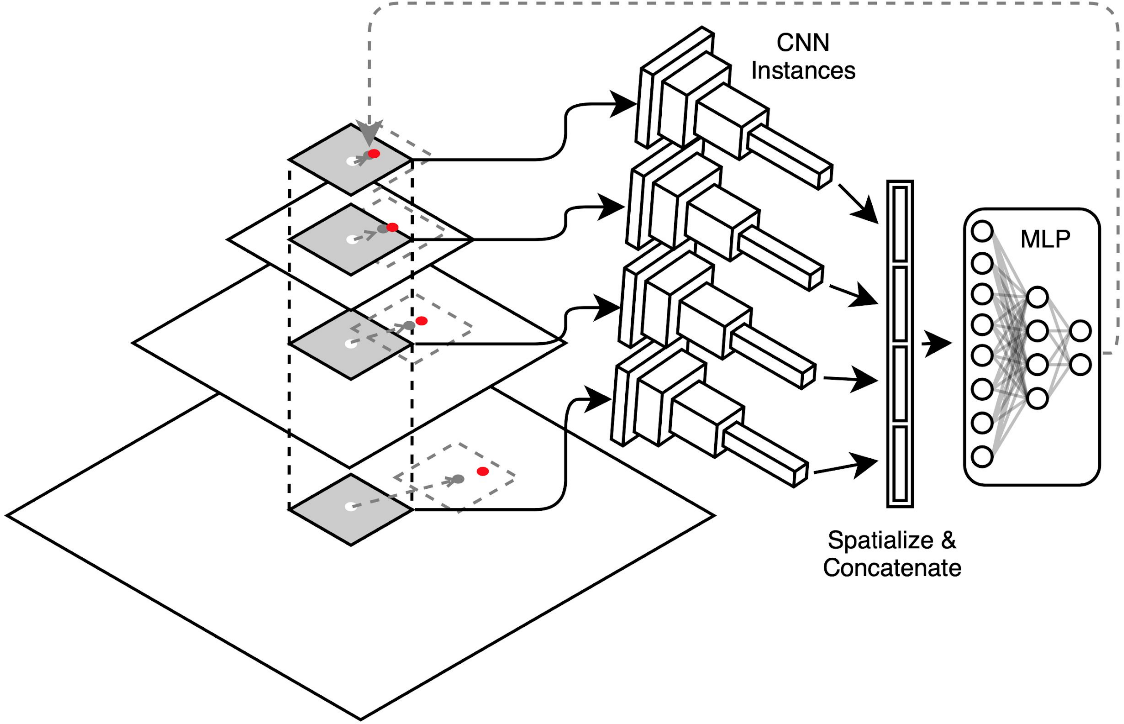
Flatten all 256 x 3 outputs into one big vector (4608-vector for 6 levels), feed it to MLP.

MLP: 4608 -> 512 -> 128 -> 2. Relu activations.

Predicts an error (grey dashed arrow) between our previous estimate (white dot) and the ground truth (red dot).

We can then repeat this whole process from the new estimate (grey dot).

No backpropogation through time.



Training

The initial estimate is taken from a normal distribution centered on the landmark location.

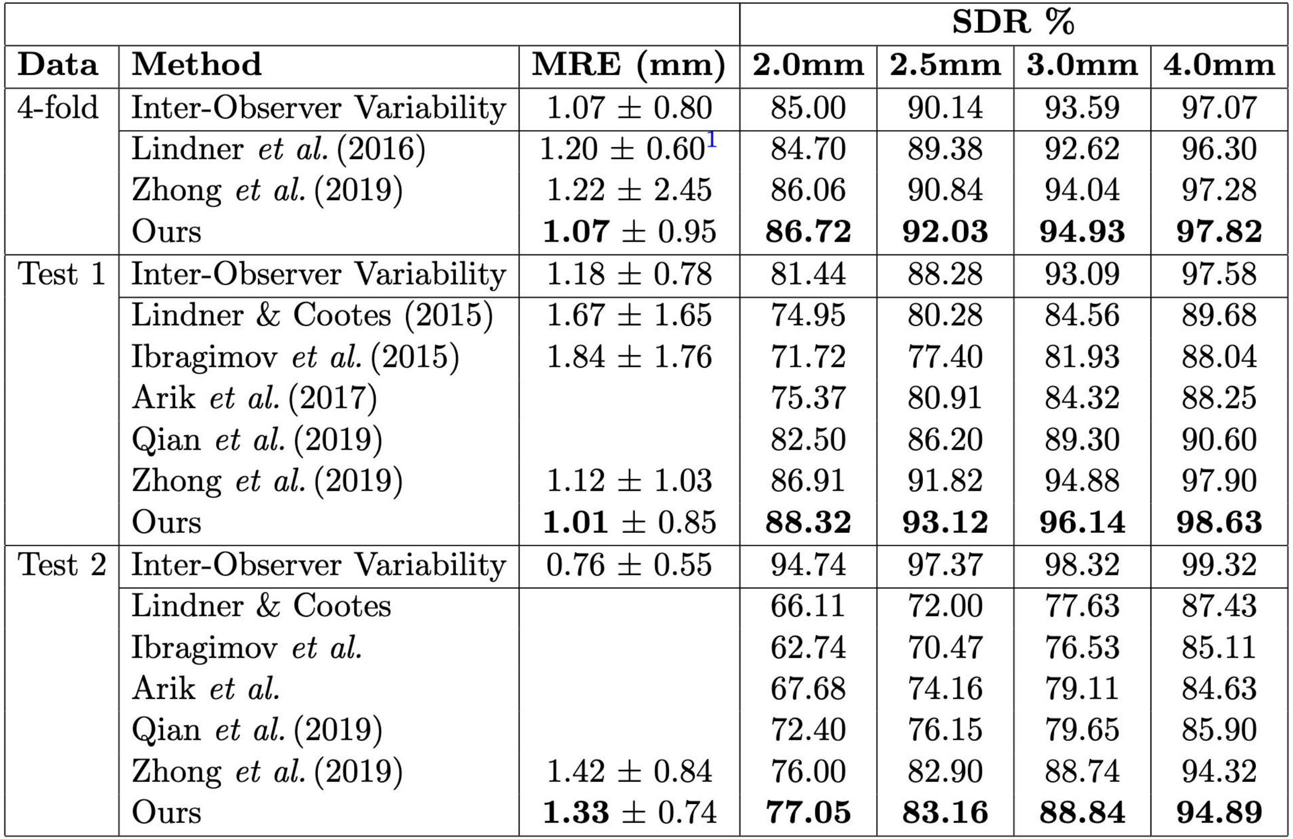
One network trained for each landmark.

Trained with ADAM for 20 epochs at lr 1e-4, and 20 epochs at lr 1e-5.

Results:

SDR: Successful Detection Ratio at various thresholds.

MRE: Mean Radial Error.



			SDR %			
Data	**Method**	**MRE (mm)**	**2.0mm**	**2.5mm**	**3.0mm**	**4.0mm**
4-fold	Inter-Observer Variability	1.07 ± 0.80	85.00	90.14	93.59	97.07
	Lindner *et al.* (2016)	1.20 ± 0.60[1]	84.70	89.38	92.62	96.30
	Zhong *et al.* (2019)	1.22 ± 2.45	86.06	90.84	94.04	97.28
	Ours	**1.07** ± 0.95	**86.72**	**92.03**	**94.93**	**97.82**
Test 1	Inter-Observer Variability	1.18 ± 0.78	81.44	88.28	93.09	97.58
	Lindner & Cootes (2015)	1.67 ± 1.65	74.95	80.28	84.56	89.68
	Ibragimov *et al.* (2015)	1.84 ± 1.76	71.72	77.40	81.93	88.04
	Arik *et al.* (2017)		75.37	80.91	84.32	88.25
	Qian *et al.* (2019)		82.50	86.20	89.30	90.60
	Zhong *et al.* (2019)	1.12 ± 1.03	86.91	91.82	94.88	97.90
	Ours	**1.01** ± 0.85	**88.32**	**93.12**	**96.14**	**98.63**
Test 2	Inter-Observer Variability	0.76 ± 0.55	94.74	97.37	98.32	99.32
	Lindner & Cootes		66.11	72.00	77.63	87.43
	Ibragimov *et al.*		62.74	70.47	76.53	85.11
	Arik *et al.*		67.68	74.16	79.11	84.63
	Qian *et al.* (2019)		72.40	76.15	79.65	85.90
	Zhong *et al.* (2019)	1.42 ± 0.84	76.00	82.90	88.74	94.32
	Ours	**1.33** ± 0.74	**77.05**	**83.16**	**88.84**	**94.89**

Discussion

Good use of transfer learning! CNNs must learn to be somewhat scale invariant because of foreshortening, and our multi-scale approach uses that property despite all images being at same scale.

Has a sort of built-in data augmentation (each image is exploded into many crops at many scales), which might help explain good performance even on relatively small data.

Interesting to note that while 10 iterations worked best at train time, as few as 3 iterations is enough at inference time, suggesting the efficacy of 10 iterations at train time is due to the resulting sampling density.

Thanks!