Holistic Image Understanding with Deep Learning and Dense Random Fields

Shuai Zheng, Anurag Arnab, Bernardino Romera-Paredes

Outline

- Introduction to Semantic segmentation
- Conditional Random Fields as Recurrent Neural Networks
- Higher Order Conditional Random Fields in Deep Neural Networks
- Recurrent Instance Segmentation

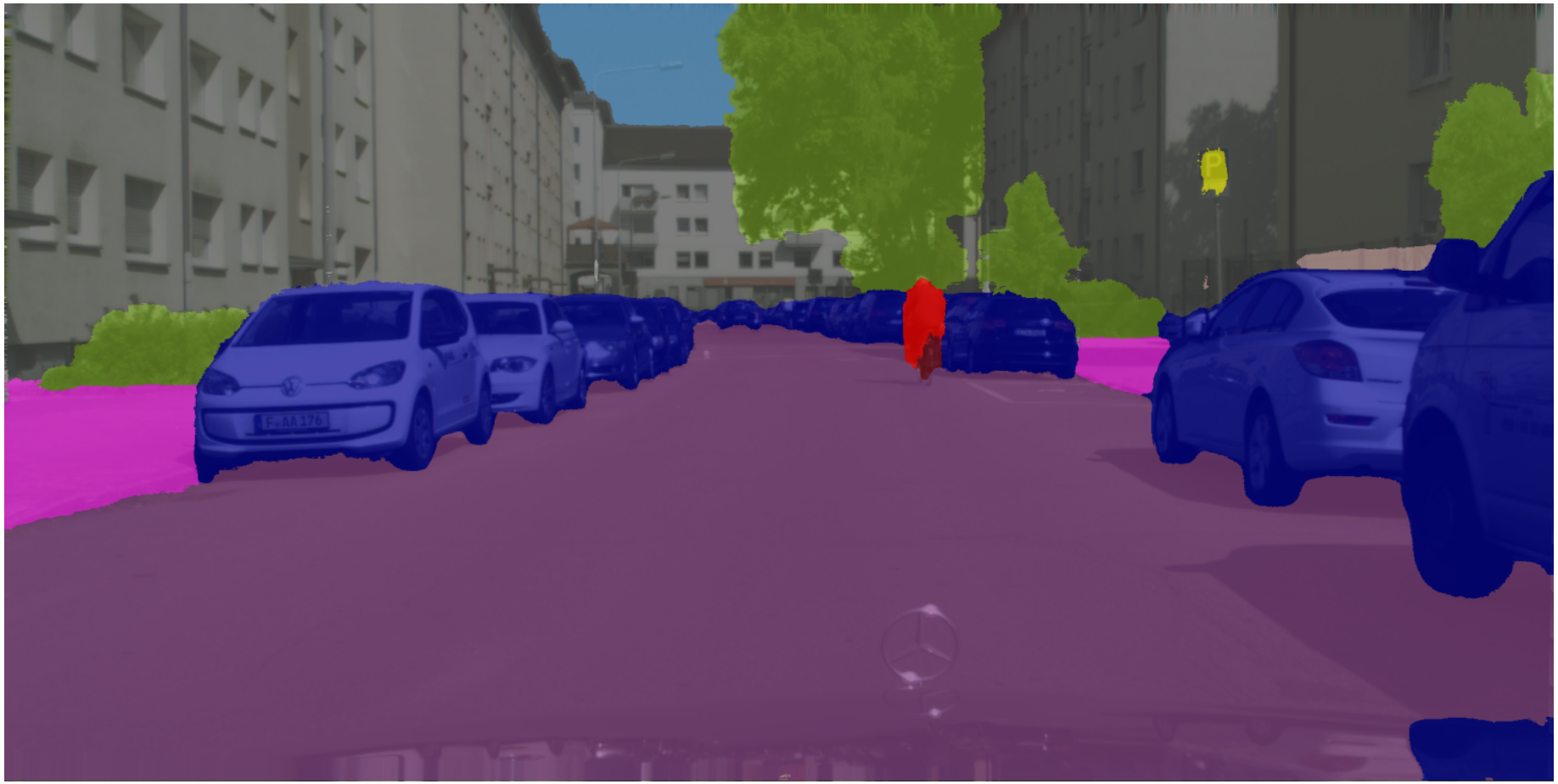
Semantic Segmentation

Duygulu ECCV 02; He et al. CVPR 04; Shotton ECCV 06.

Why Semantic Segmentation?

OX·SIGHT

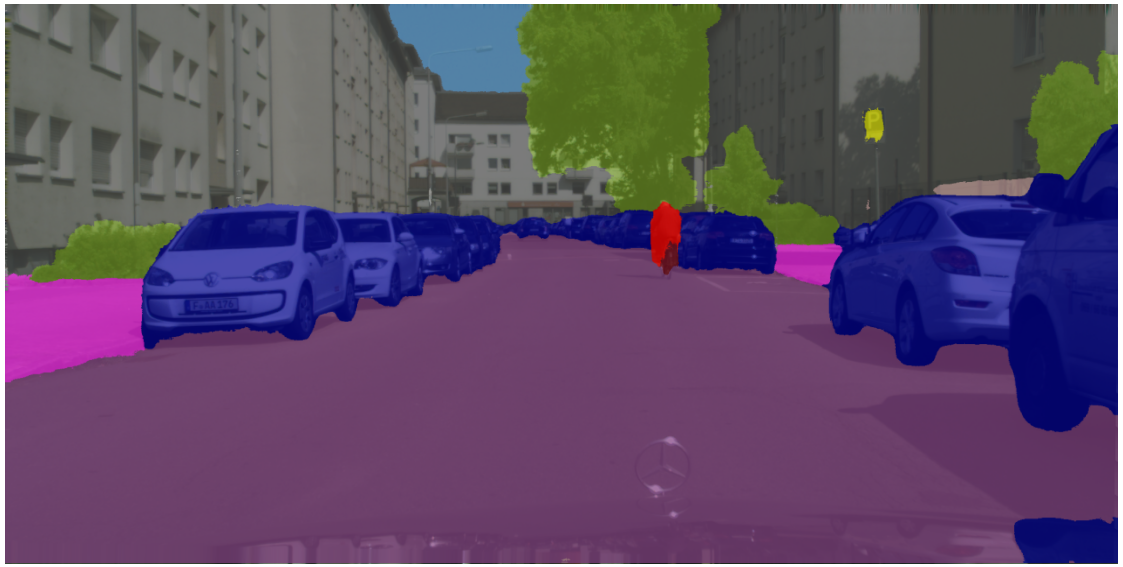
Why Semantic Segmentation?

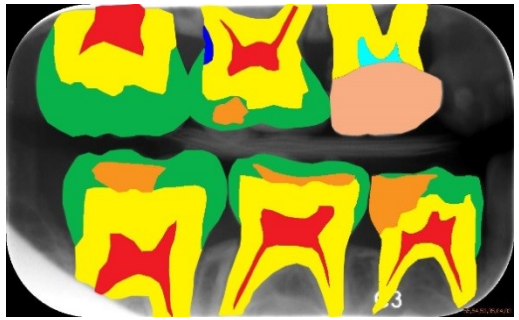


Results of CRFasRNN on validation image of Cityscape dataset.

Why Semantic Segmentation?

- Autonomous navigation
- Assisting the partially sighted
- Medical diagnosis
- Image editing

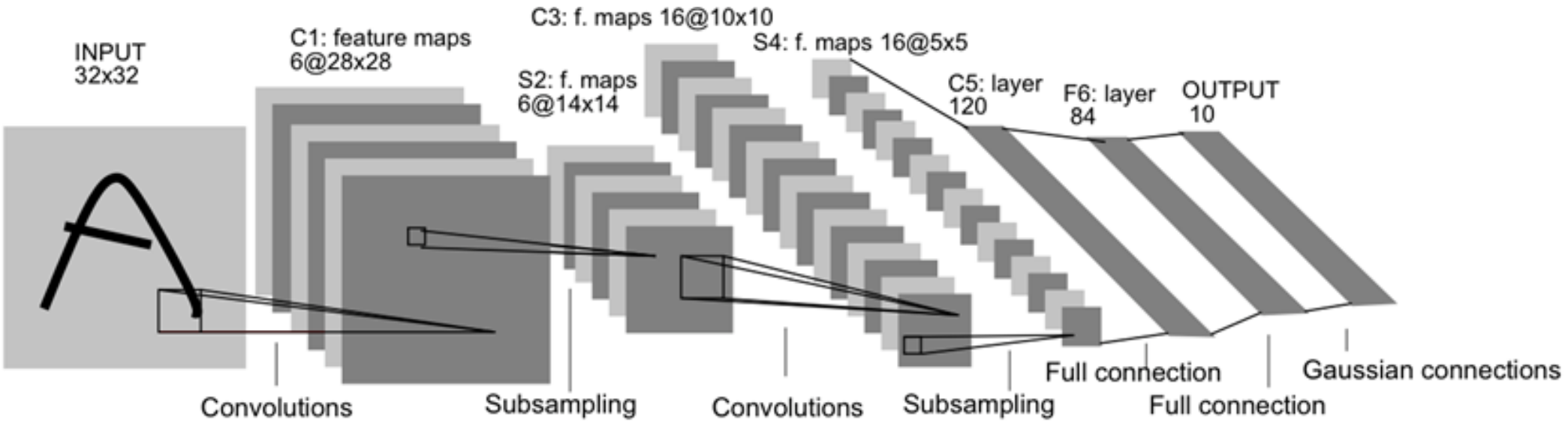




(Clockwise from top) [1] Cityscapes Dataset. [2] ISBI Challenge 2015, dental x-ray images. [3] Royal National Institute of the Blind

But How?

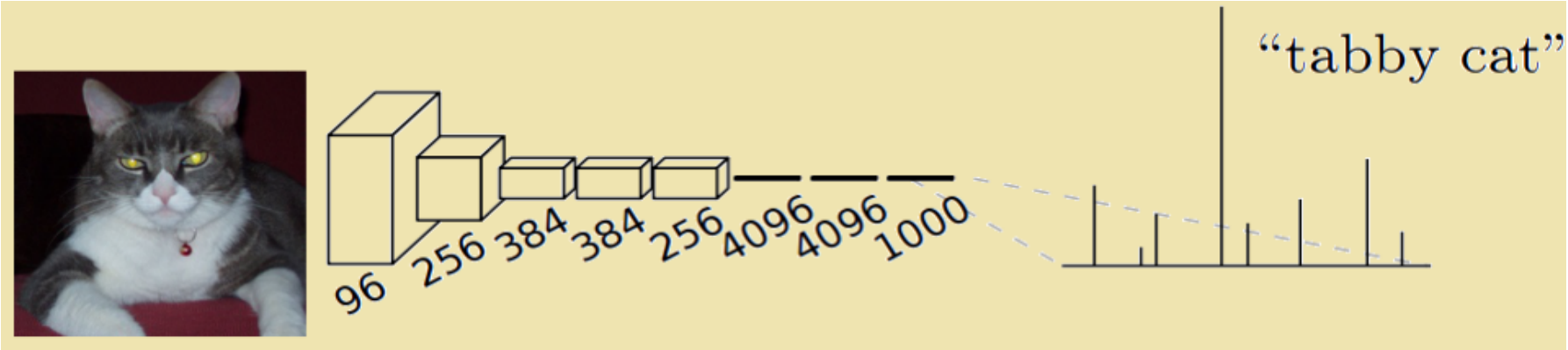
- Convolutional neural networks are successful at learning a good representation of visual inputs.
- How to make use of CNNs for structured output prediction problems?



Yann LeCun's Optical Character recognition system in 1990s.

CNN for Pixelwise Labelling

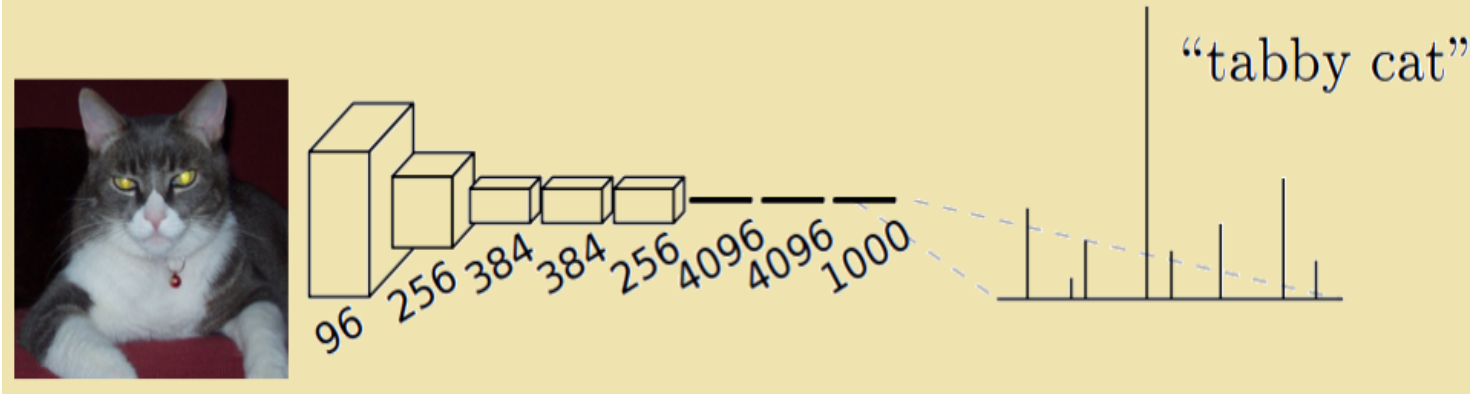
- Traditional Convolutional Neural Networks



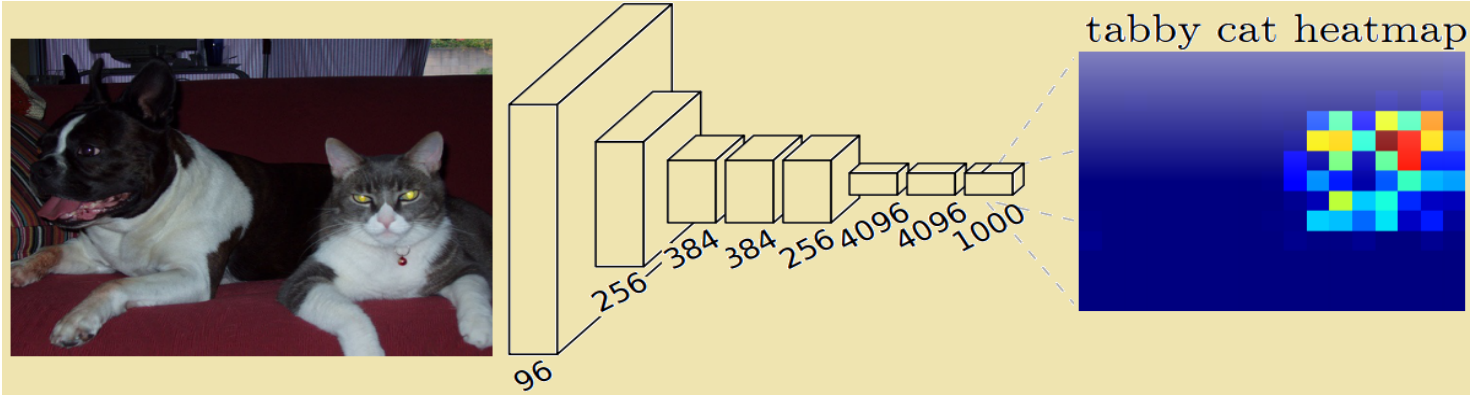
Alex's CNN model won the image classification task in ImageNet Challenge 2012 (ECCVW 2012).

CNN for Pixelwise Labelling

- Traditional Convolutional Neural Networks

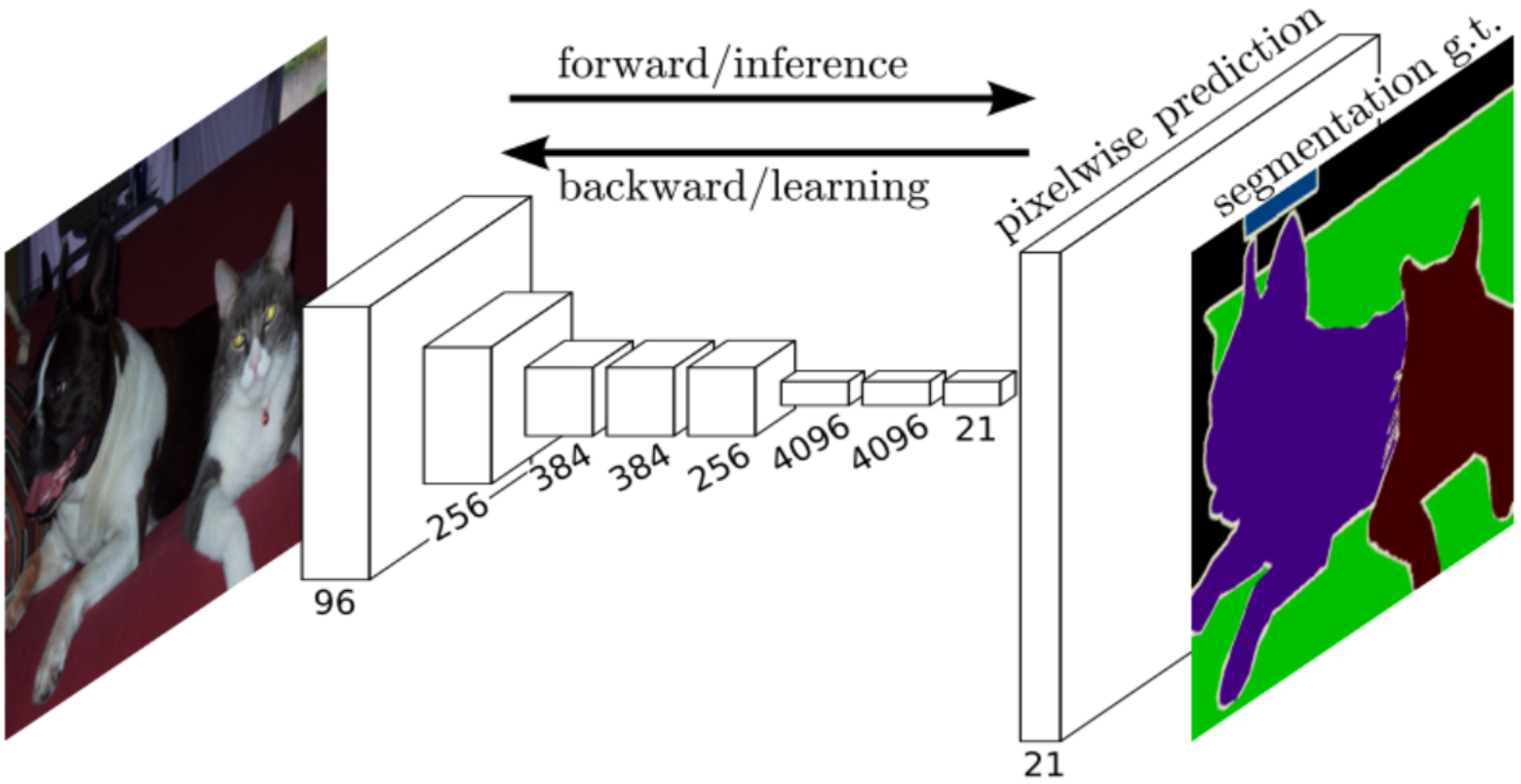


- Fully convolutional networks



Jon Long*, Evan Shelhamer*, Trevor Darrell. CVPR 2015.

Fully Convolutional Networks



Jon Long*, Evan Shelhamer*, Trevor Darrell. CVPR 2015.

Fully Convolutional Networks

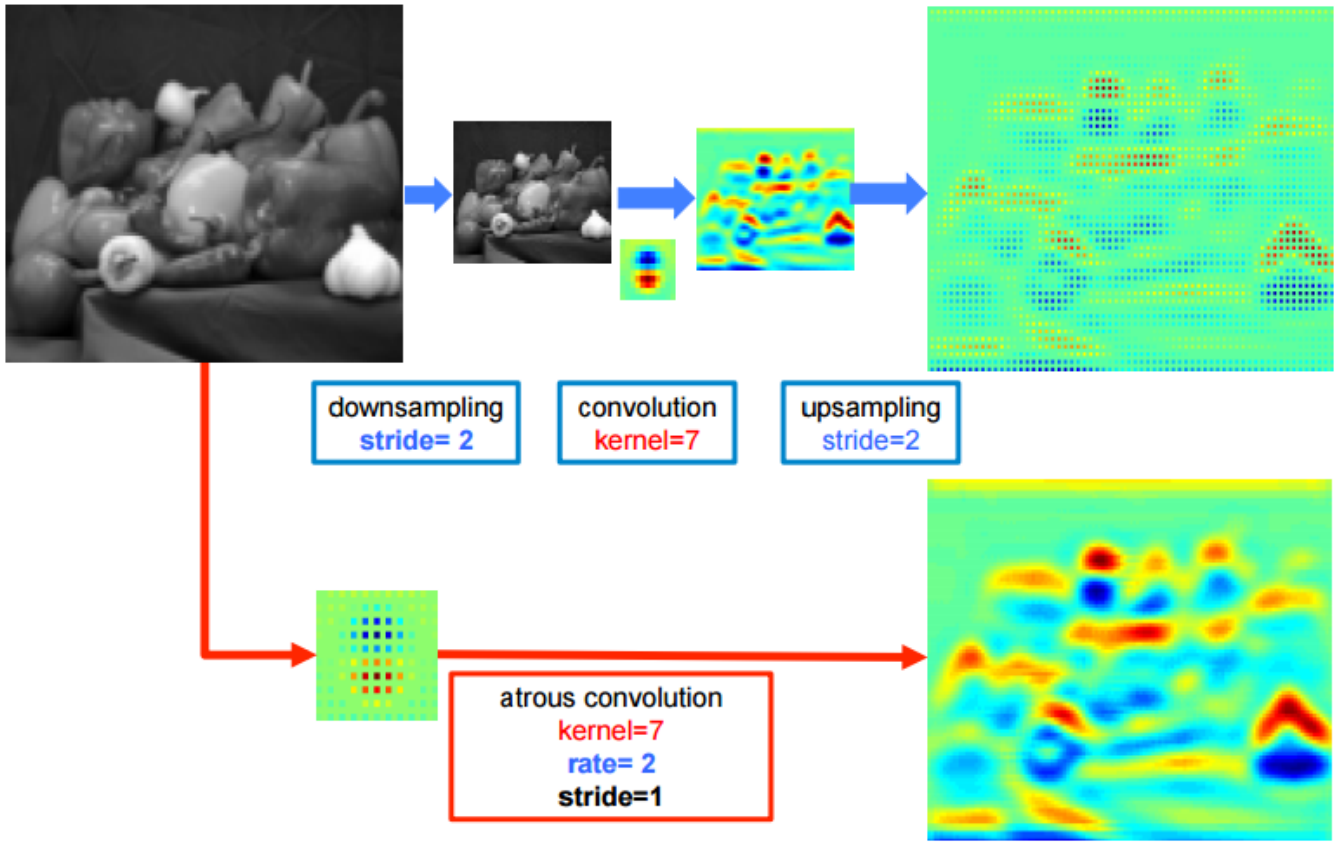
+ Significantly improved the state-of-the-art in semantic segmentation.
- Poor object delineation: e.g. spatial/color consistency neglected.

Image

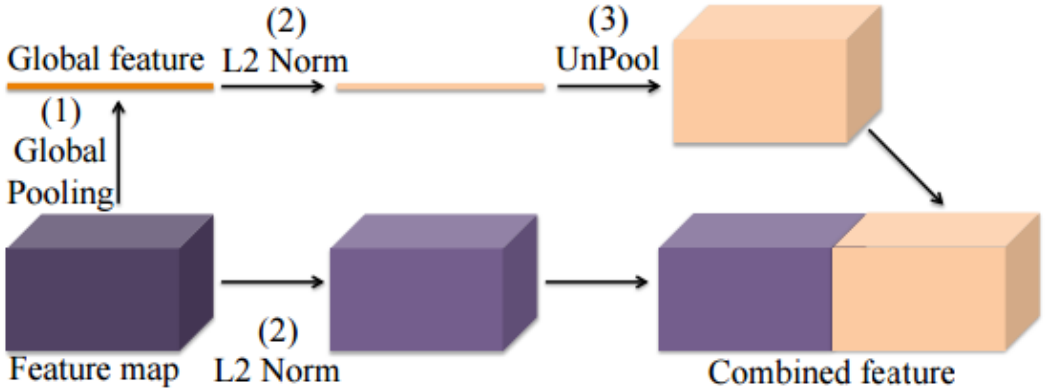
FCN Results

Ground truth

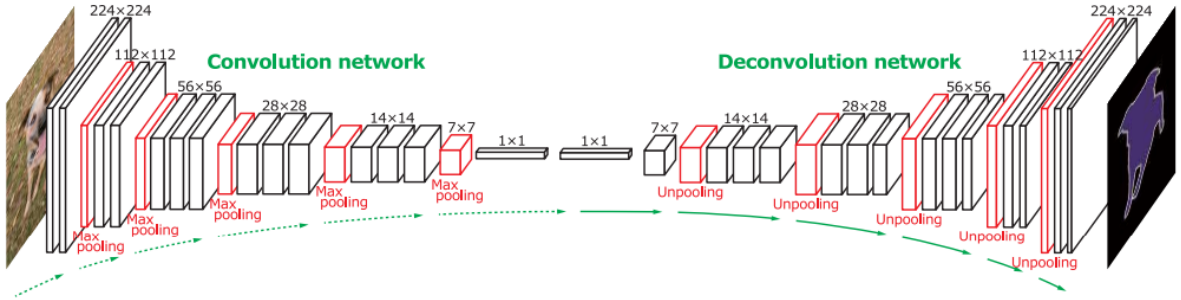
How to improve FCN?



1) Atrous Convolution (A.K.A. dilated conv)



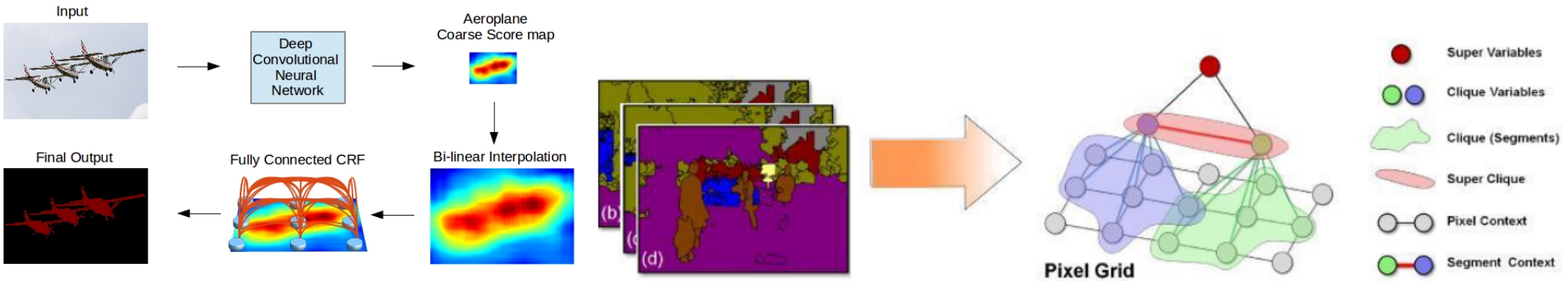
2) ParseNet

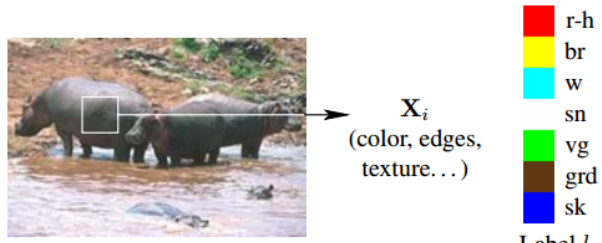


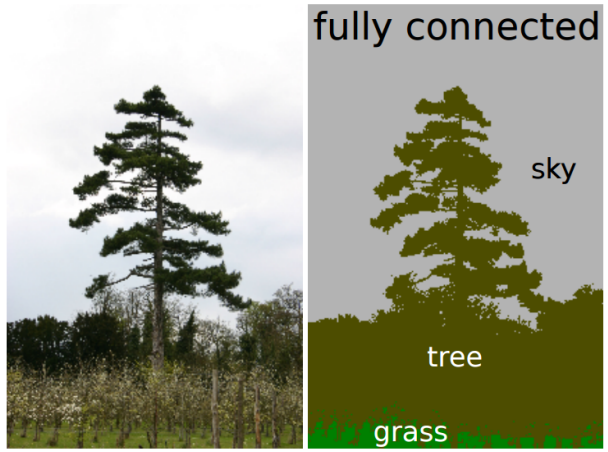
3) Unpooling (Deconvolution) Network

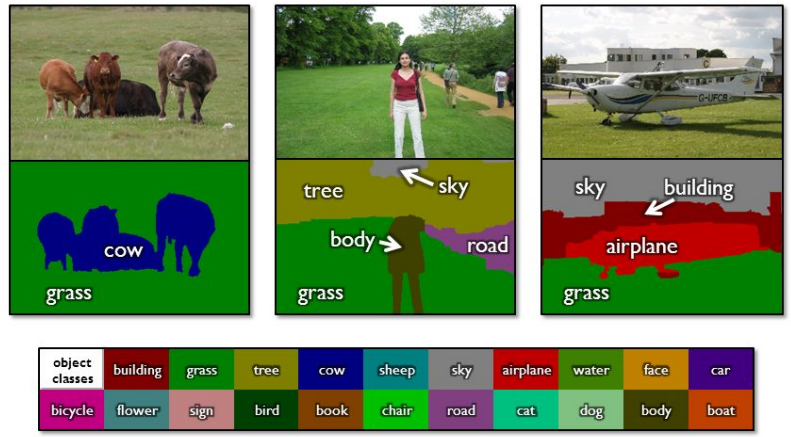
Chen et al. ICLR 2015. Liu et al. ICLR 2016. Noh et al. ICCV 2015.

Post-processing with CRFs









He et al. CVPR 04; Shotton et al. ECCV 06; Ladicky et al. ICCV 09; Krähenbühl et al. NIPS 2011; Chen et al.ICLR 15.

How to improve FCN?

1. Atrous Convolution (also known as dilated convolution)
2. ParseNet
3. Unpooling (Deconvolution) Network
4. Post Processing with Conditional Random Fields
5. End-to-end training with Conditional Random Fields

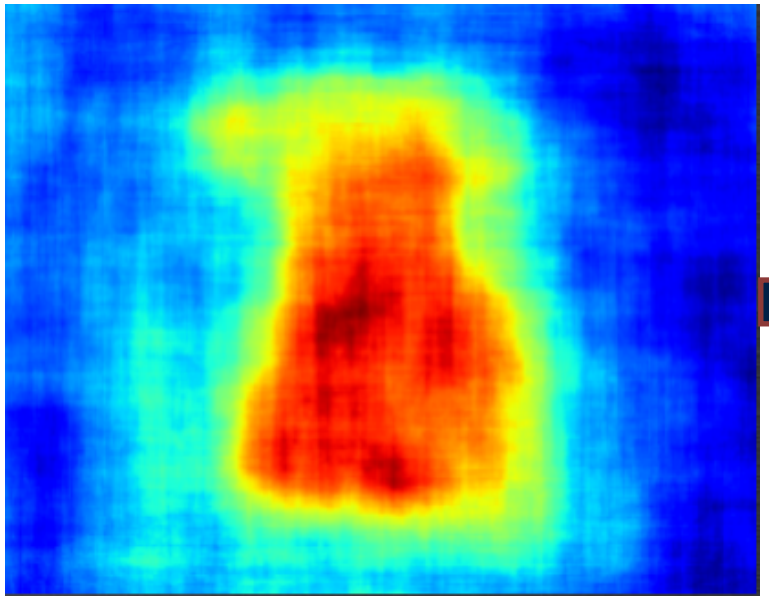
Chen et al. ICLR 2015. Liu et al. ICLR 2016. Noh et al. ICCV 2015.

How to improve FCN?

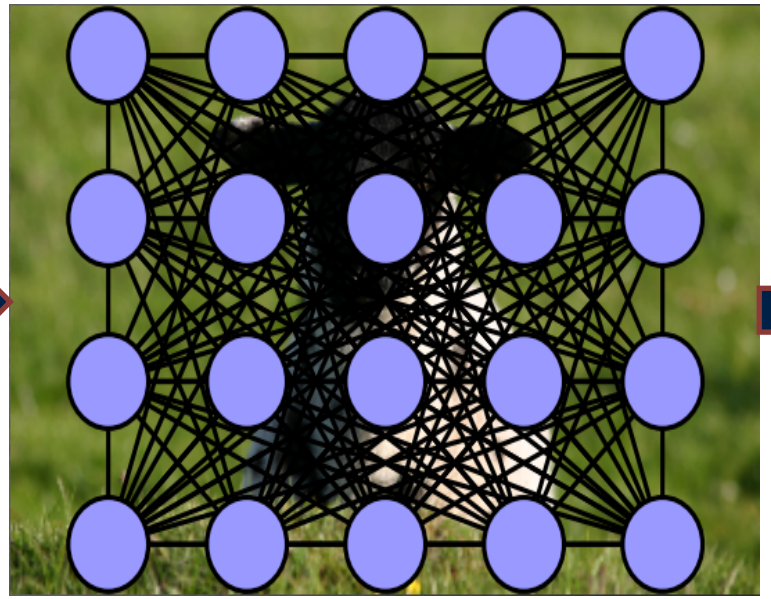
1. Atrous Convolution (also known as dilated convolution)
2. ParseNet
3. Unpooling (Deconvolution) Network
4. Post Processing with Conditional Random Fields
5. End-to-end training with Conditional Random Fields

Conditional (Markov) Random Fields

- A CRF can account for contextual information in the image



Coarse output from the pixel-wise classifier

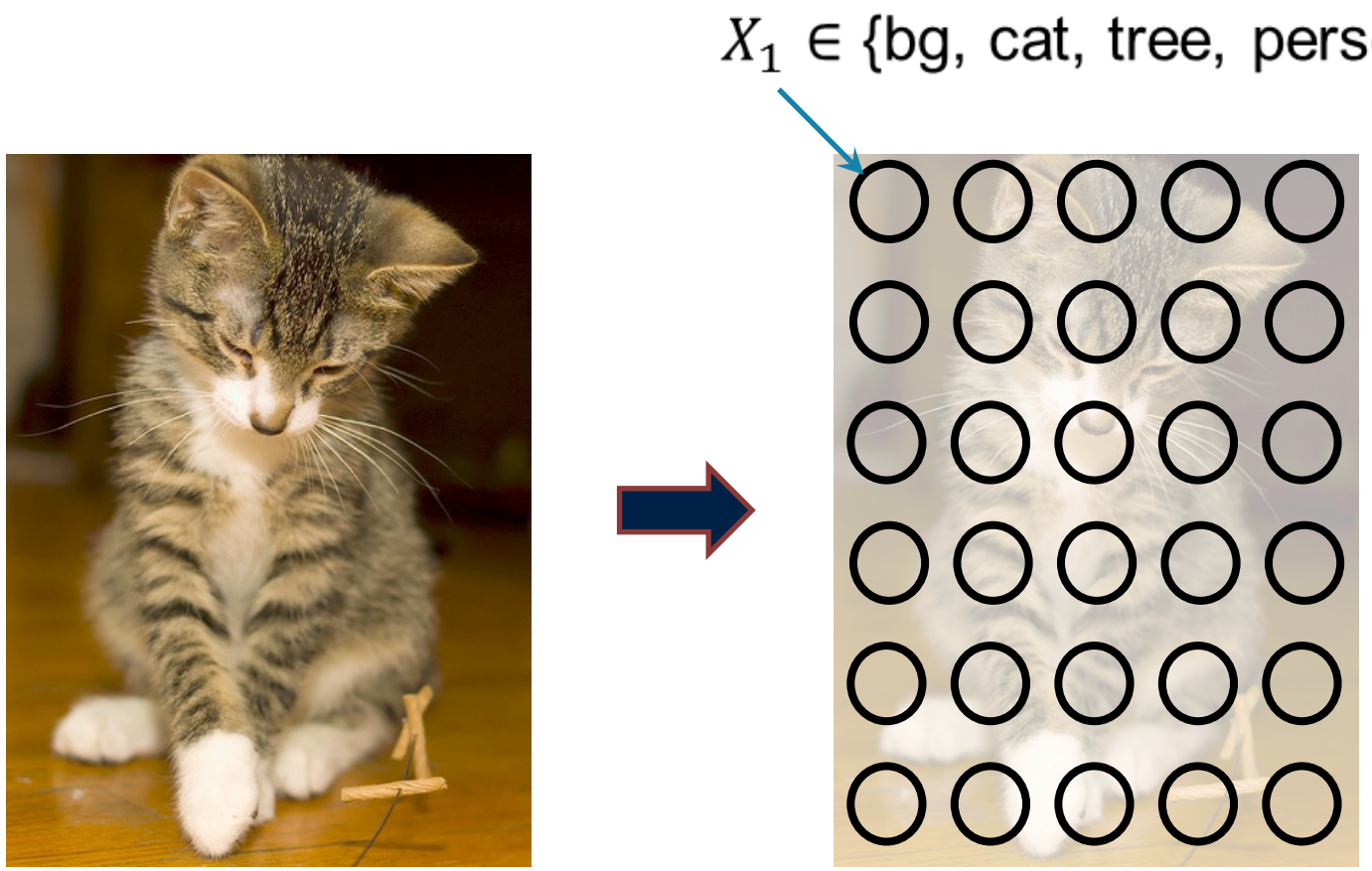


CRF modelling

Output after the CRF inference

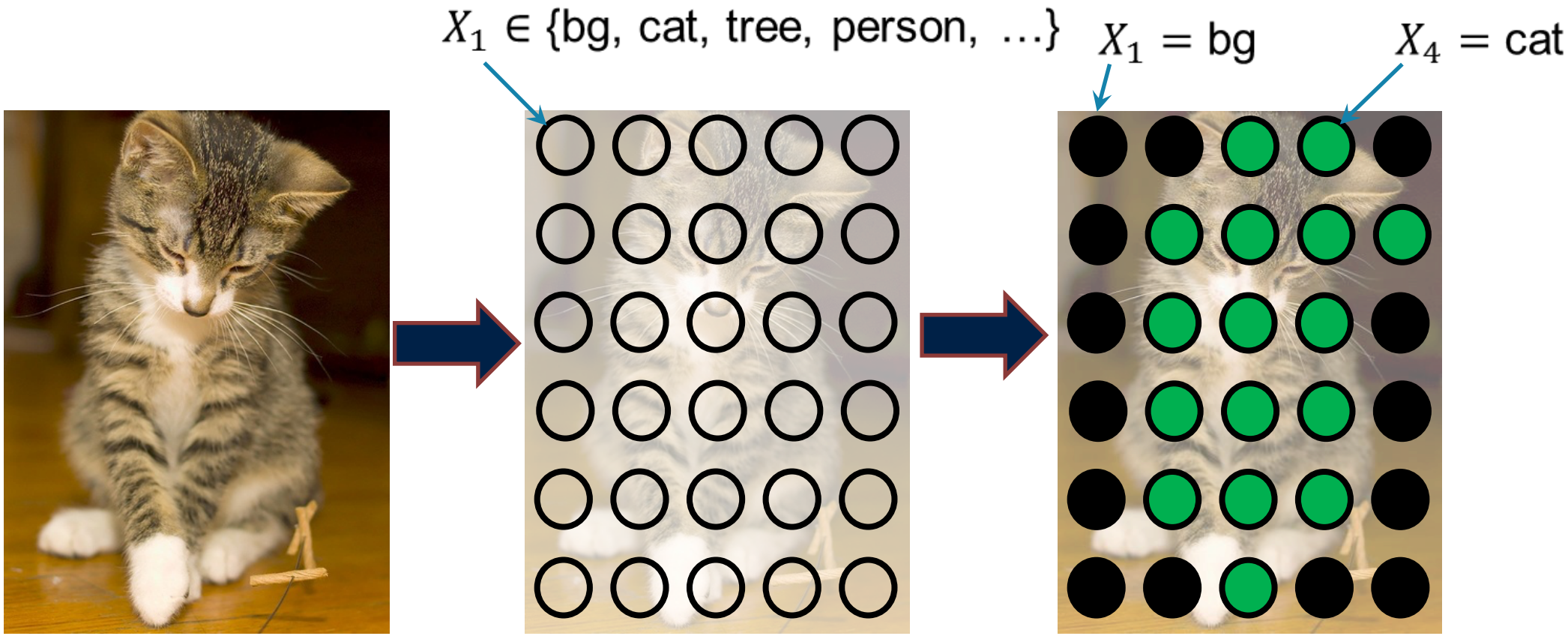
Shotton et al. 2006; Ladicky et al. 2009; Krähenbühl et al. 2011 & 2013.

Pixel-wise labelling problems



- Define a discrete random variable X_i for each pixel i
- Each X_i can take a value from the label set
- Connect random variables to form a random field

Pixel-wise labelling problems



Segmentation results: Most probable assignment given the image.

Example of other pixel labelling problems

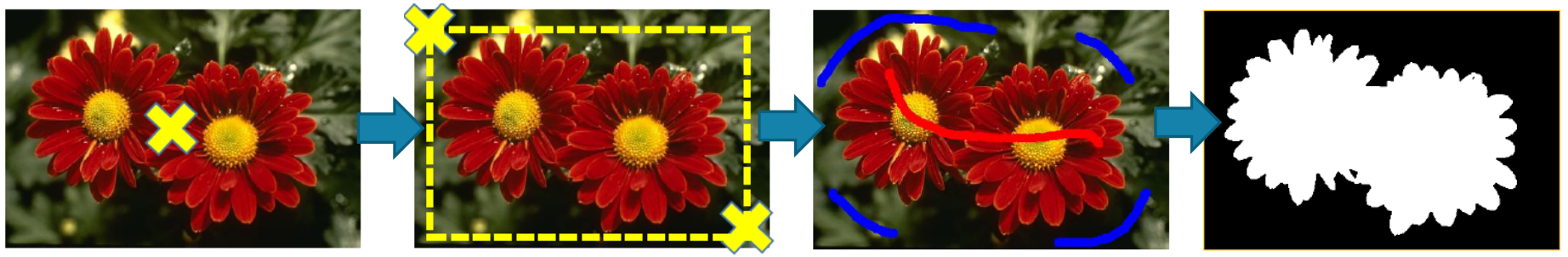


Image Segmentation

Pose estimation

Instance segmentation

Rother et al. ACM TOG 2004; Seguin et al. CVPR 2013.

Solving pixel-wise labeling as an energy minimization problem

The probability for the pixel-wise assignment

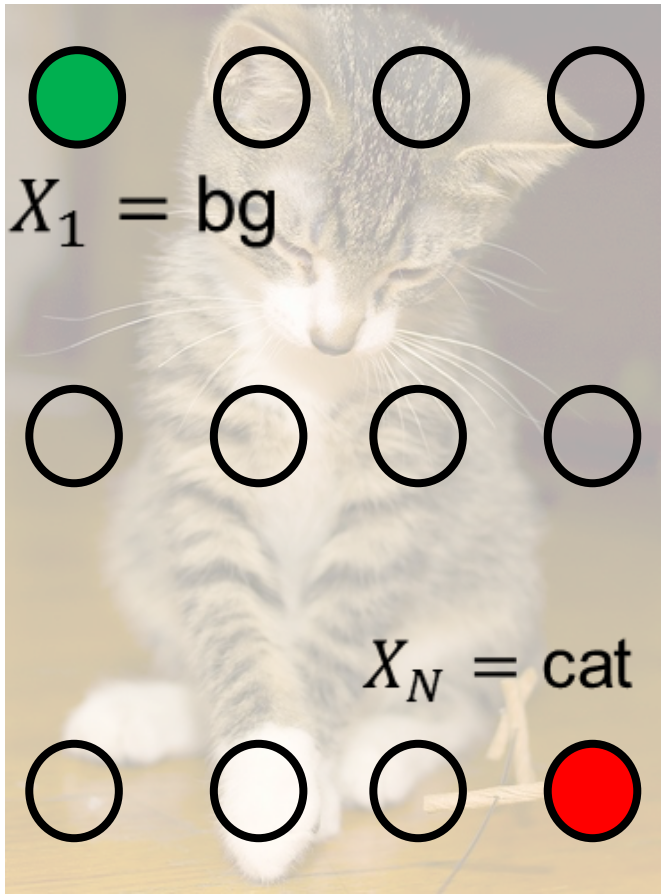
$$\Pr(X_1 = x_1, \ldots, X_N = x_n | I) = \Pr(\boldsymbol{X} = x | \boldsymbol{I})$$

Let energy function be E, we have

$$\Pr(X_1 = x \mid I) = \exp(-E(x|\boldsymbol{I}))$$

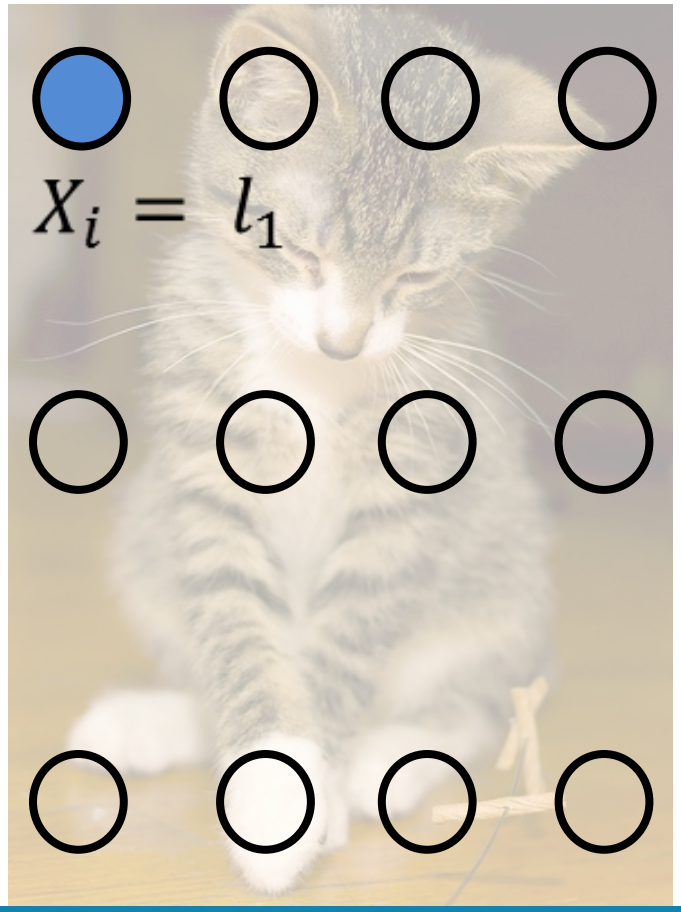
Maximum a Posteriori (MAP) Inference is equivalent to energy minimization

$$\arg\max_x \Pr(X = x|\boldsymbol{I}) \rightarrow \arg\min_x E(x|\boldsymbol{I})$$



Energy

$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

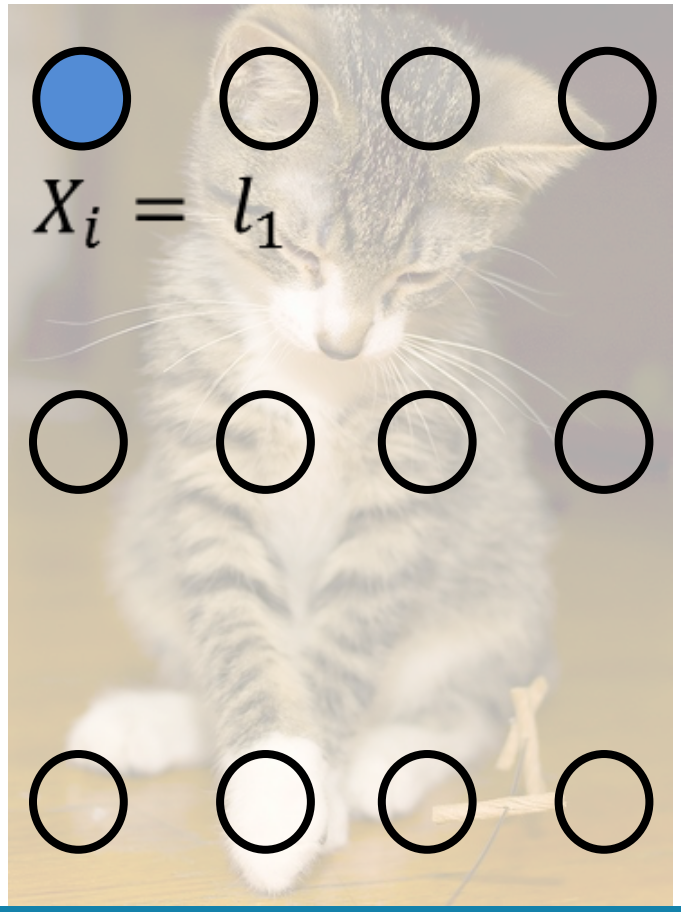


Energy

$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

Unary energy

- $Cost(X\_i = l\_1)$ =?

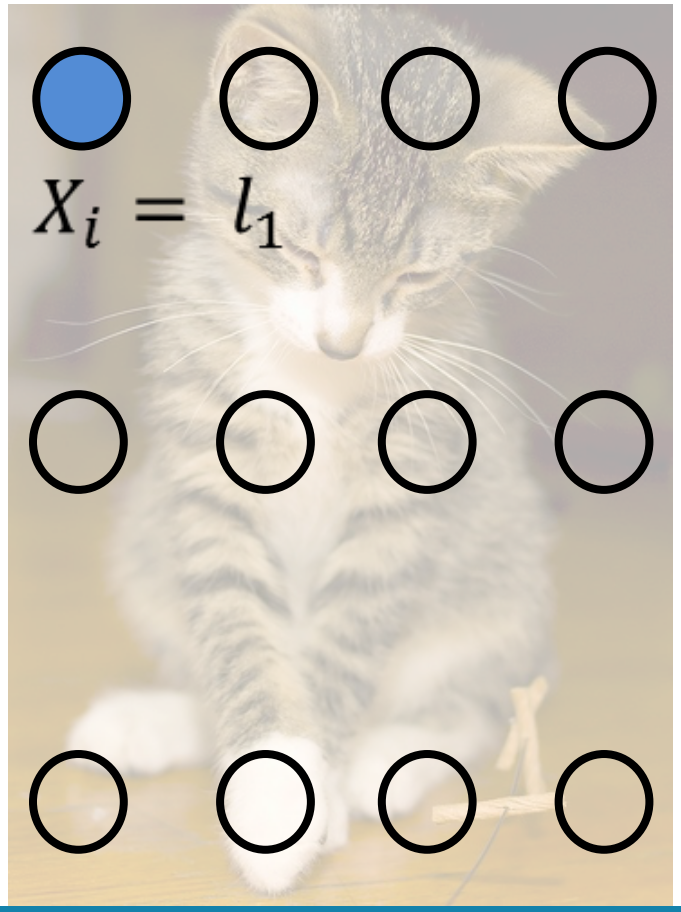


Energy

$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

Unary energy

- $Cost(X\_i = l\_1)$ =?
- Your label does not agree with the initial classifier -> you pay a penalty



Energy

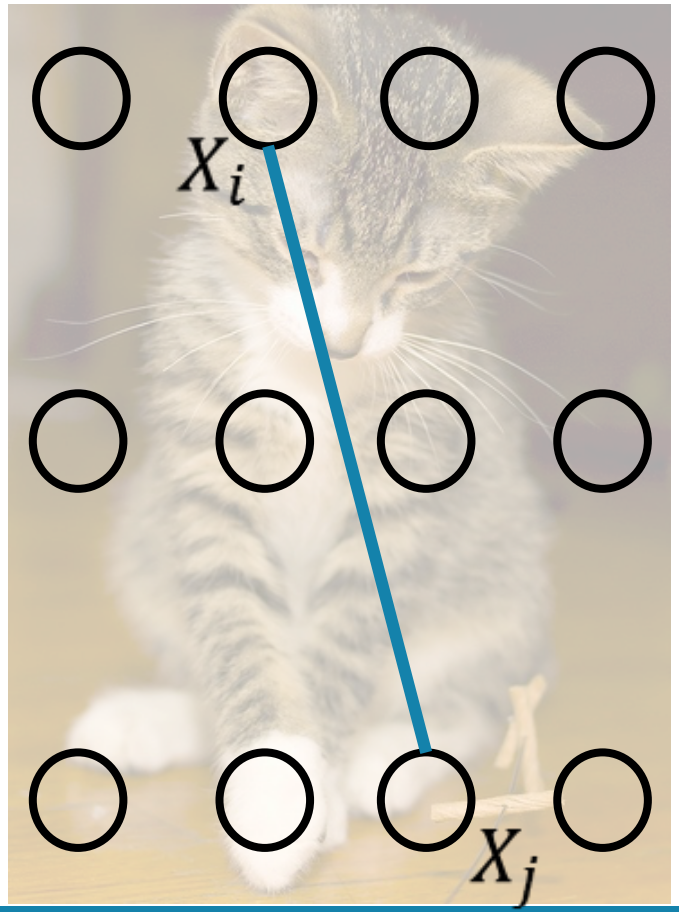
$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

Unary energy

- $Cost(X\_i = l\_1) =?$
- Your label does not agree with the initial classifier -> you pay a penalty

Pairwise energy

- $Cost(X\_i = l\_1) =?$
- You assign different labels to two very **similar** pixels -> you pay a penalty.
- How do you measure **similarity**?



Energy

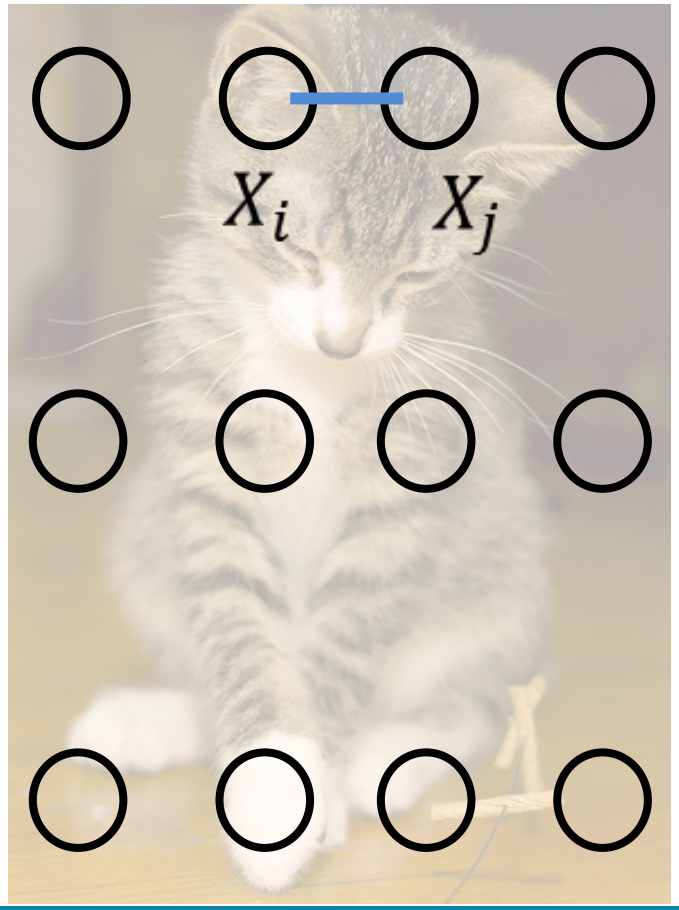
$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

Unary energy

- $Cost(X\_i = l\_1) =?$
- Your label does not agree with the initial classifier -> you pay a penalty

Pairwise energy

- $Cost(X\_i = l\_1) =?$
- You assign different labels to two very **similar** pixels -> you pay a penalty.
- How do you measure **similarity**?



Energy

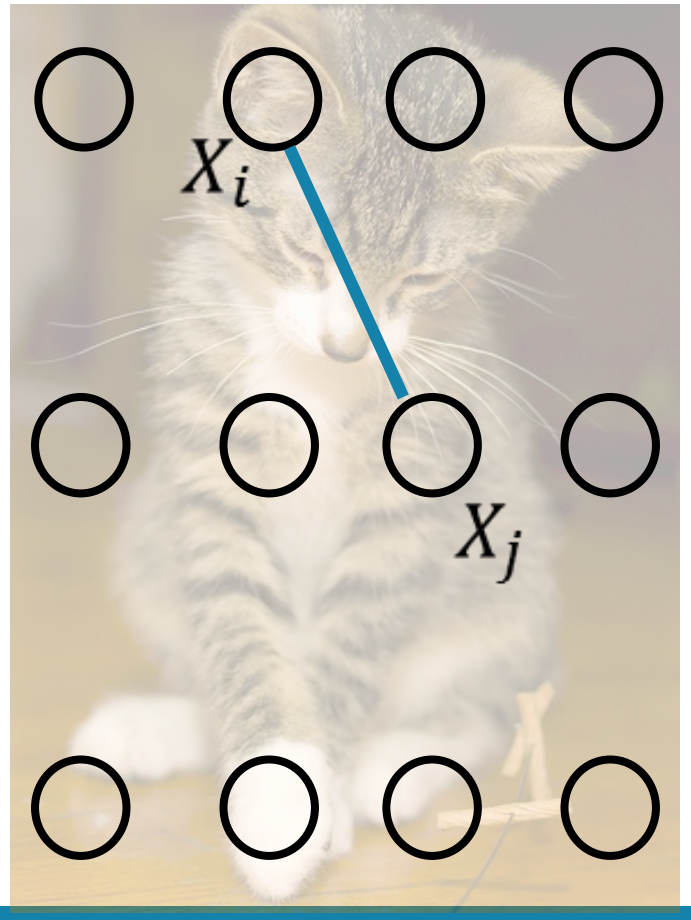
$E(x|\boldsymbol{I})$ =unary_cost + pairwise_cost

Unary energy

- $Cost(X\_i = l\_1) =?$
- Your label does not agree with the initial classifier -> you pay a penalty

Pairwise energy

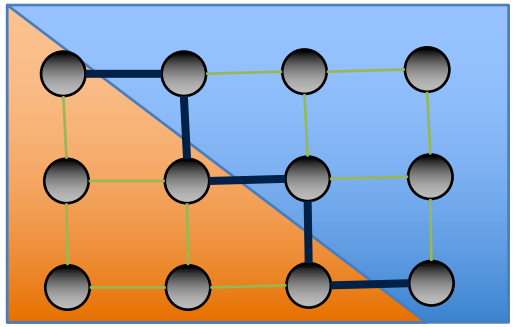
- $Cost(X\_i = l\_1) =?$
- You assign different labels to two very **similar** pixels -> you pay a penalty.
- How do you measure **similarity**?
- Color and spatial consistency



Conditional Random Fields (CRFs)

Pairwise energy

- Neighborhood
  - Option 1: considering only neighbor pixels: a grid

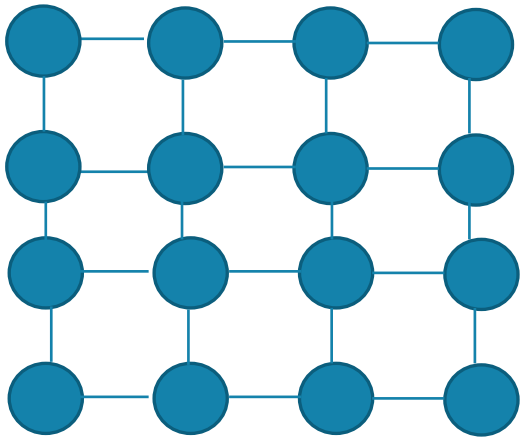


  - Fully-connected CRF

How to minimize energy function E?

Let's think about use **exact** inference to solve a "simple" problem (graph has no tree-structured):

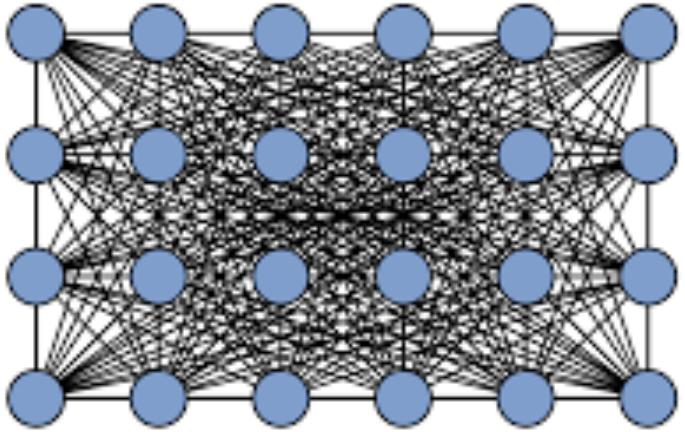
$$\arg\max_{x} \Pr(\boldsymbol{X} = x|\boldsymbol{I})$$



How to minimize energy function E?

Let's think about solving a "hard" problem using **exact** inference (graph has no tree-structure):

$$\arg\max_{x} \Pr(\boldsymbol{X} = x|\boldsymbol{I})$$



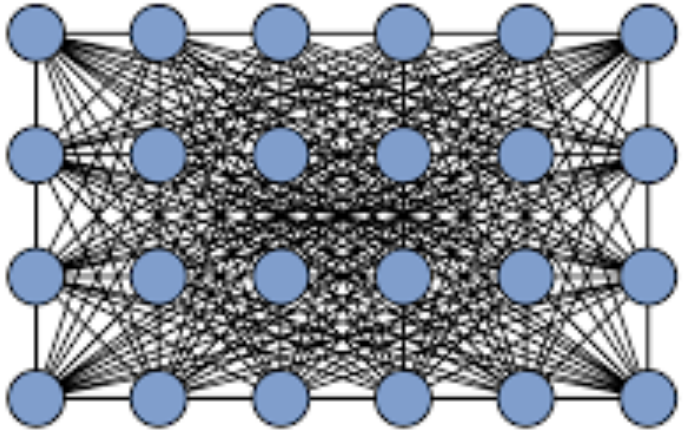

MOON
SHOT

How to minimize energy function E?

Let's think about solving a "hard" problem using **approximate** inference (graph has no tree-structure):

$$\arg\max_{x} \Pr(\boldsymbol{X} = x|\boldsymbol{I})$$

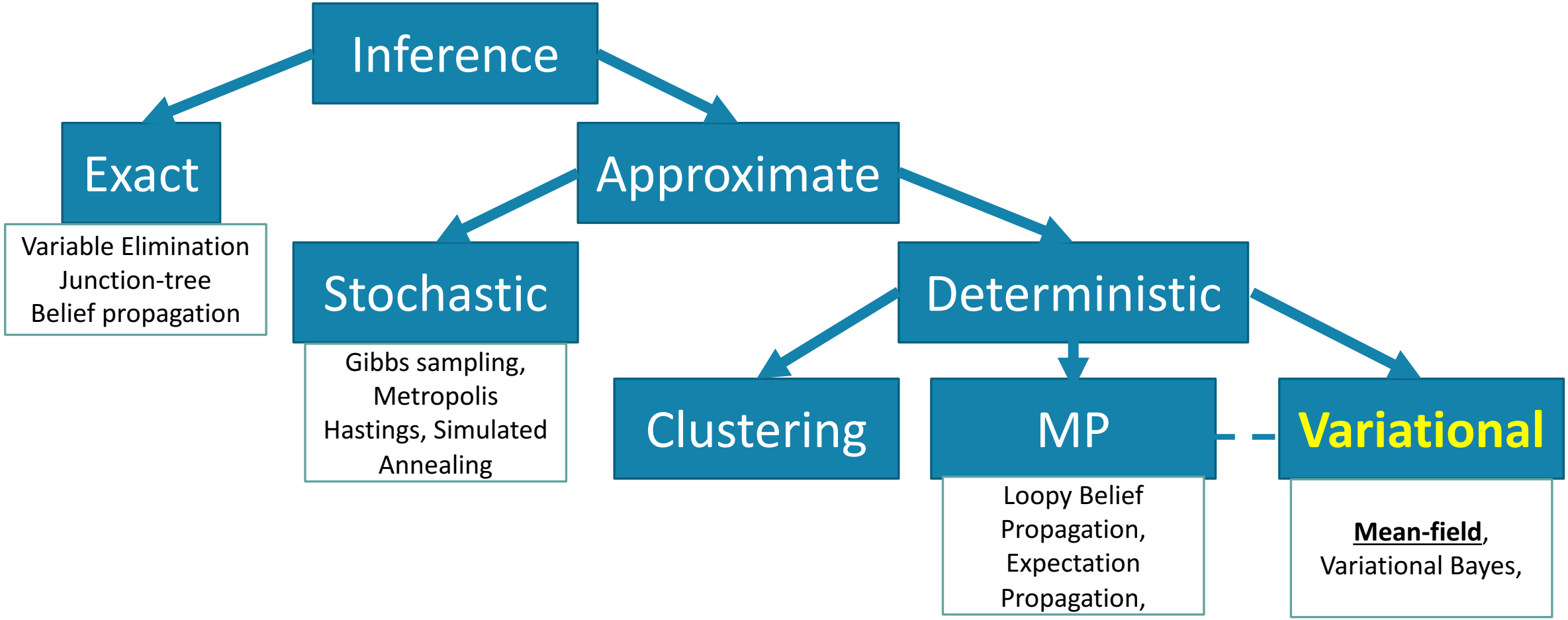
$$s.t. \Pr(\boldsymbol{X}|\boldsymbol{I}) = \frac{1}{Z}\prod_{i}^{n} \Pr(\boldsymbol{X} = x_i|\, I)$$



How to minimize energy function E?

- Exact inference
  - ✓ Accurate
  - ❑ Computational intractable, except for particular cases, e.g. tree-structured graph
- Approximate inference
  - ❑ "Less accurate"
  - ✓ Computational tractable
  - ✓ Applicable to more models

How to minimize energy function E?



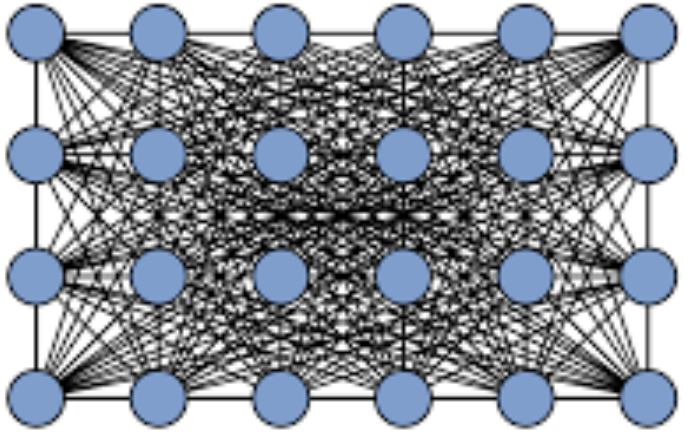
Wainwright. 2008; Koller and Friedman. 2009; Murphy 2012; Schwing and Urtasun 2015.

Fully-connected CRFs

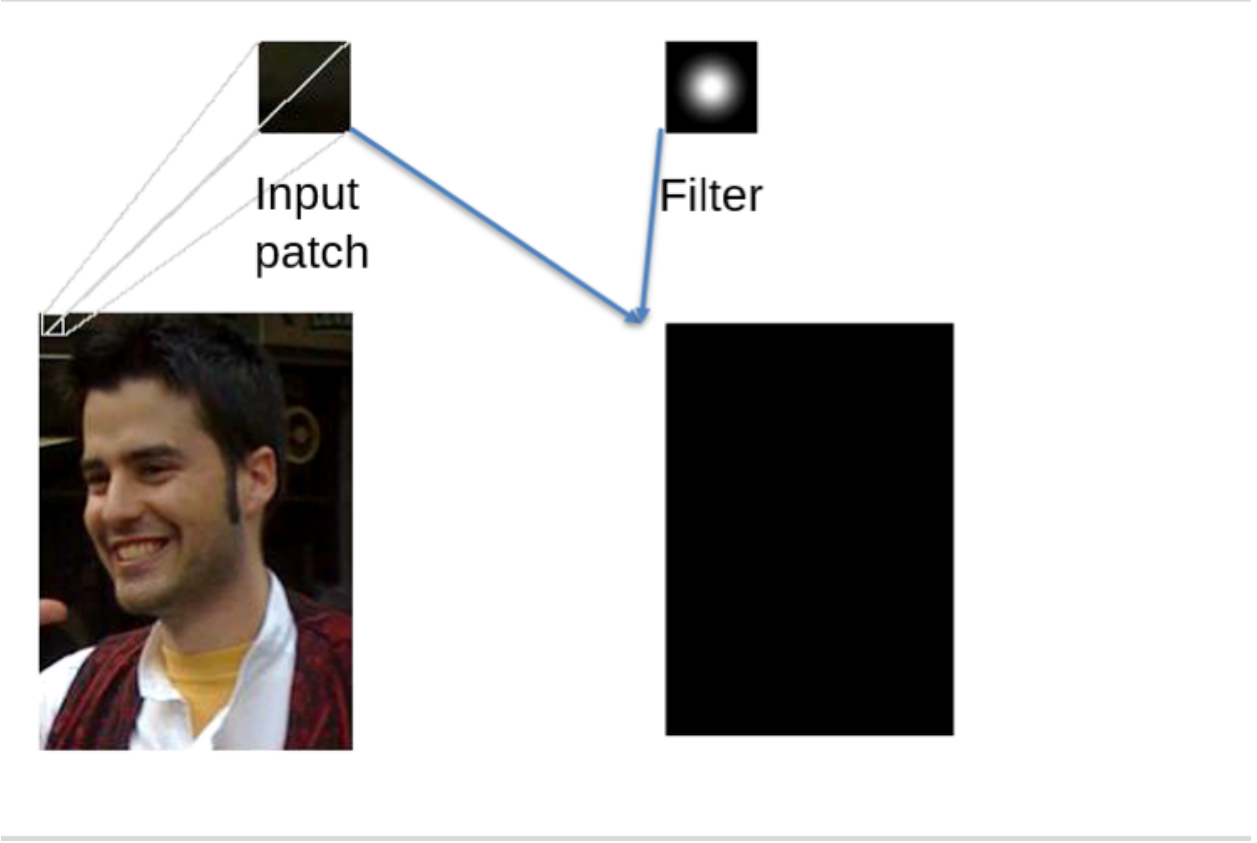
- Pairwise energies are defined for every pixel pair in the image

$$E(\boldsymbol{X}|\boldsymbol{I}) = \sum_i unary(\boldsymbol{x}_i) + \sum_{i,j} pairwise(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

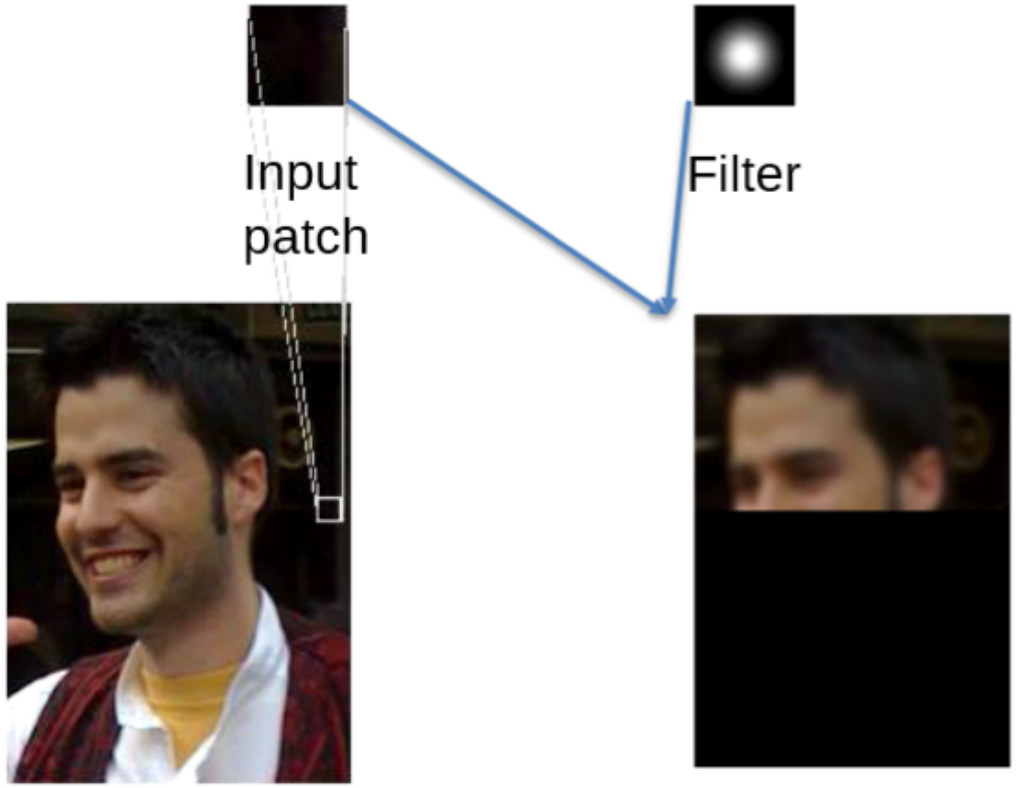
- Assuming Gaussian pairwise potentials
- Use approximate mean field inference



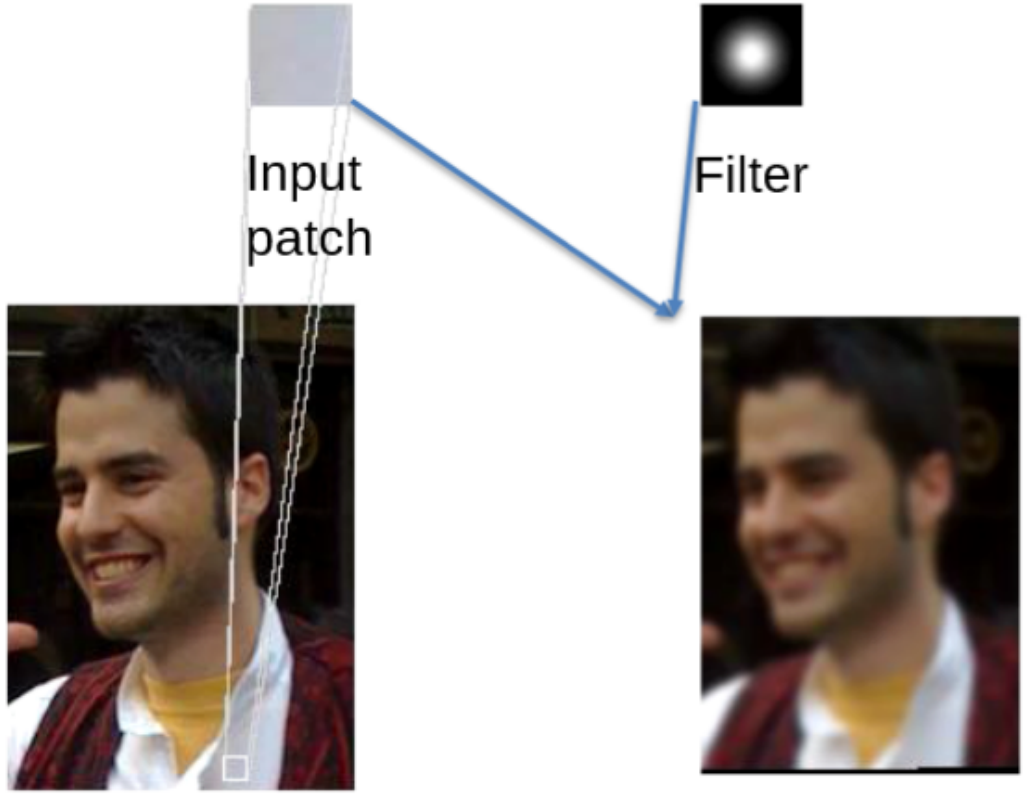
Convolution with Gaussian Filter



Convolution with Gaussian Filter



Convolution with Gaussian Filter

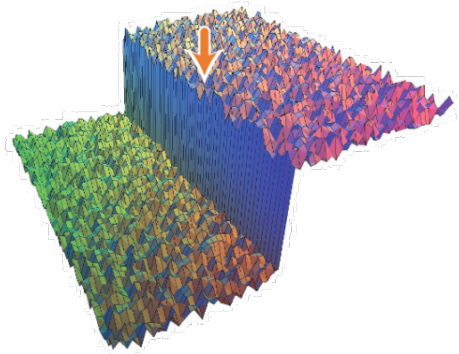


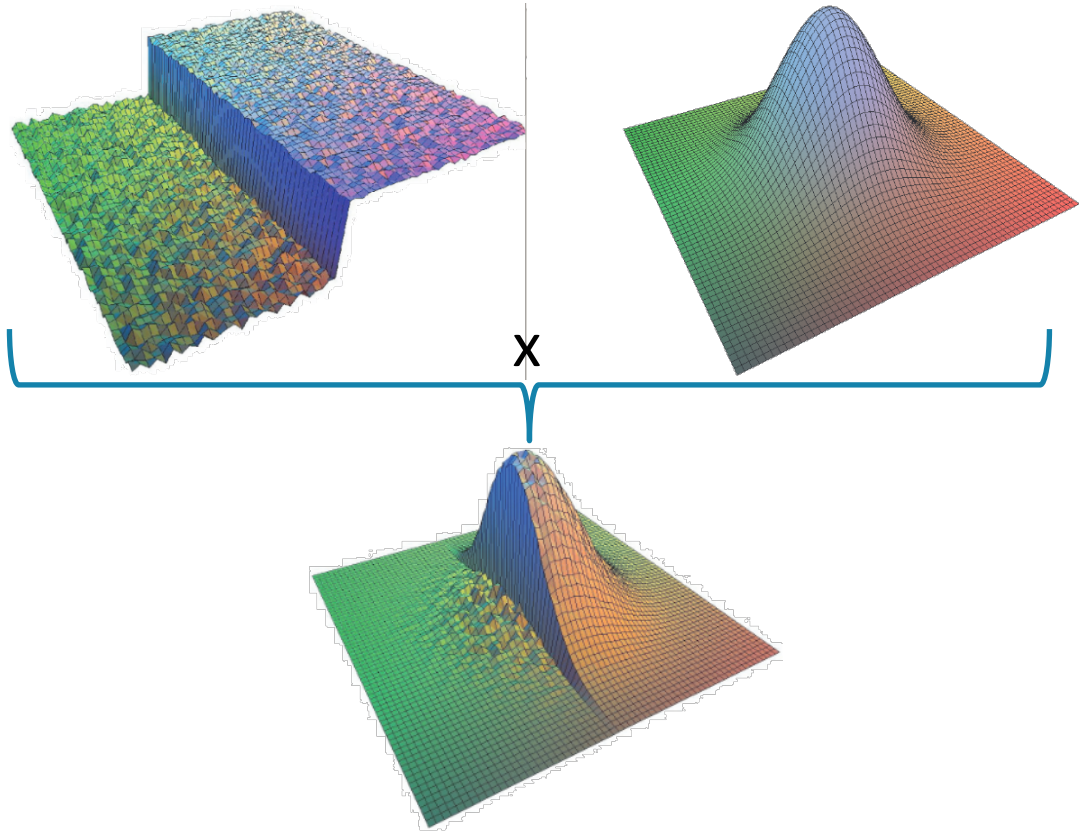
Bilateral filter

Intensity filter

Spatial filter

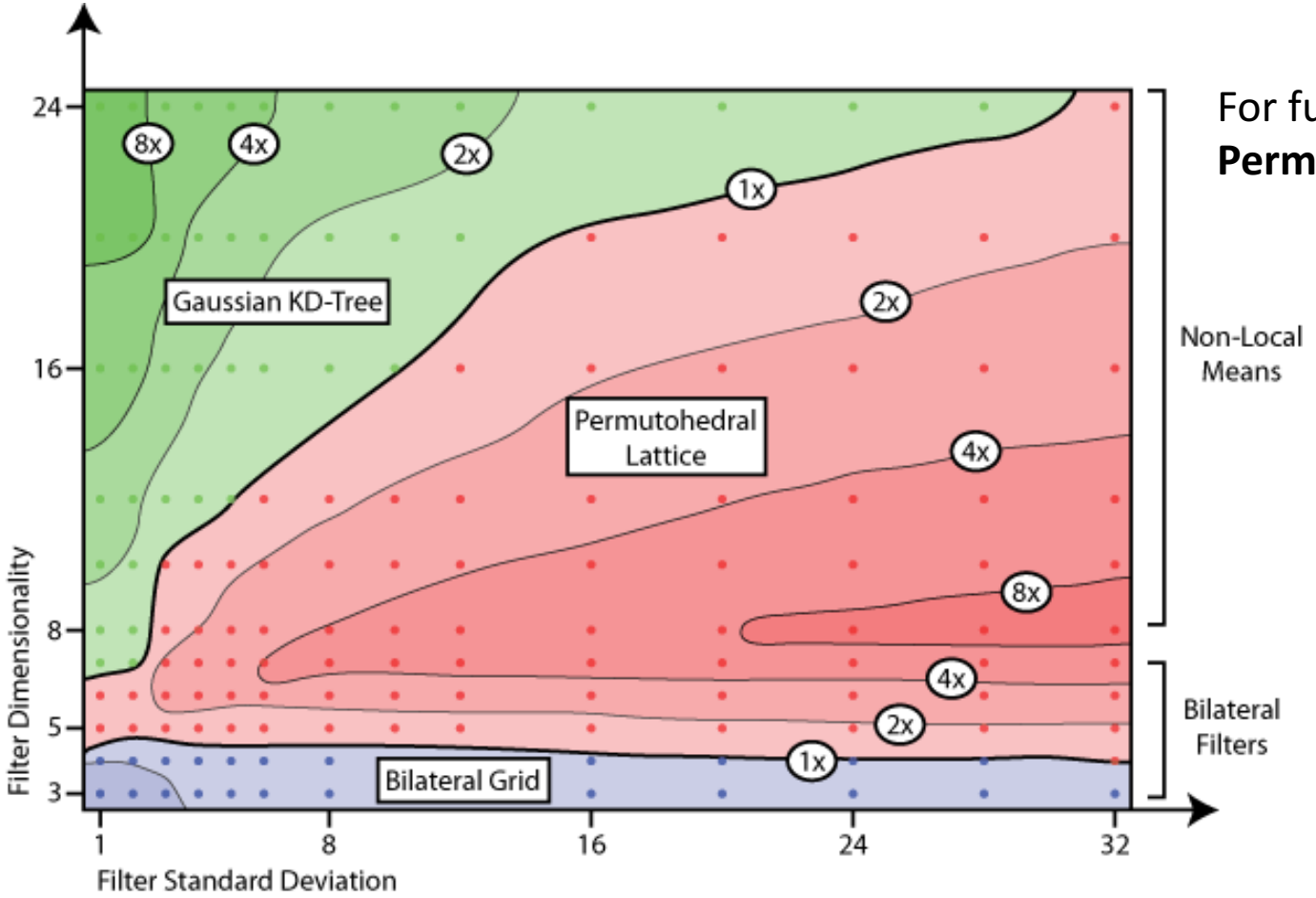
Input





Paris et al. 2008.

Bilateral filters can be fast

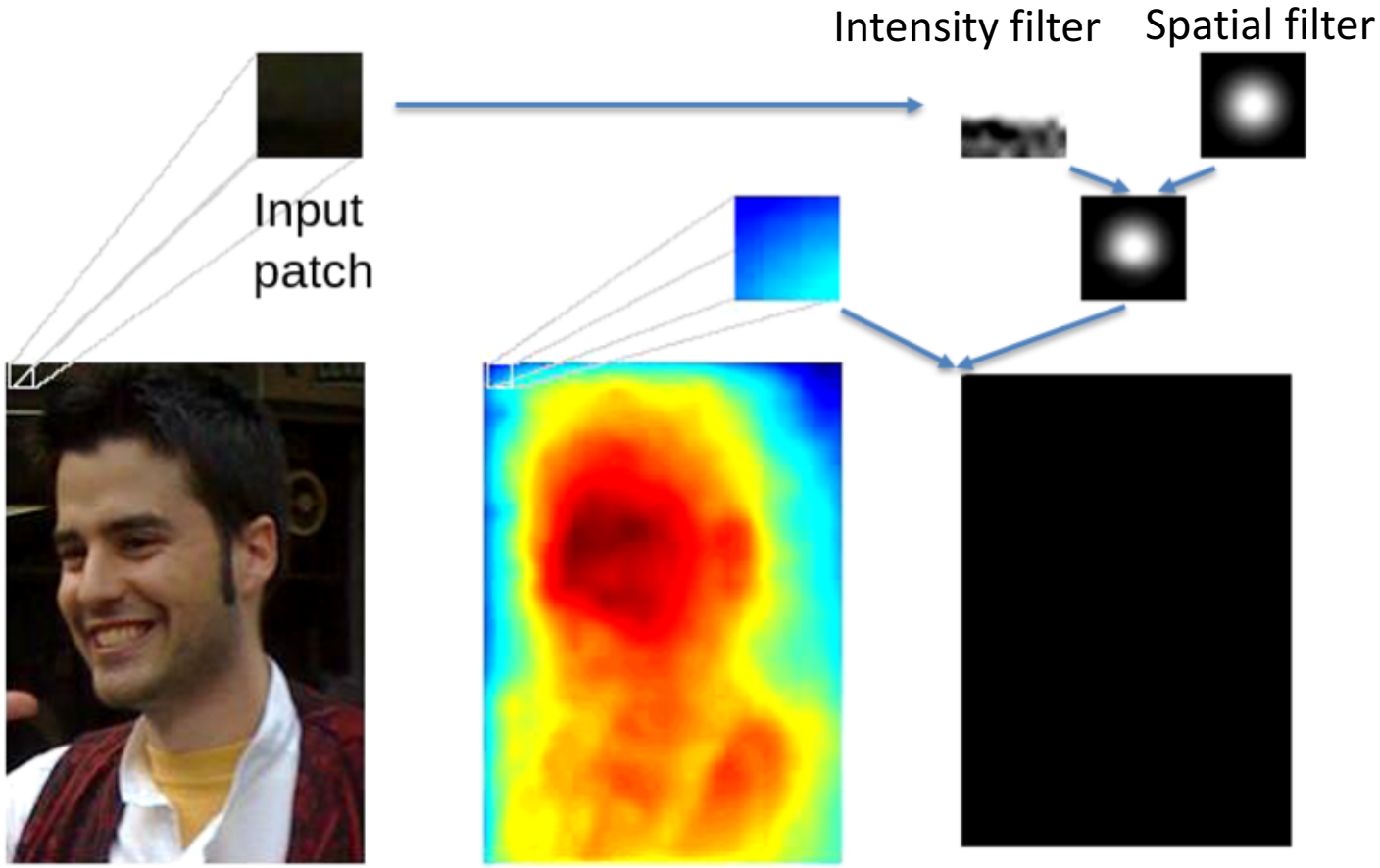


For fully-connected CRFs:
Permutohedral lattice

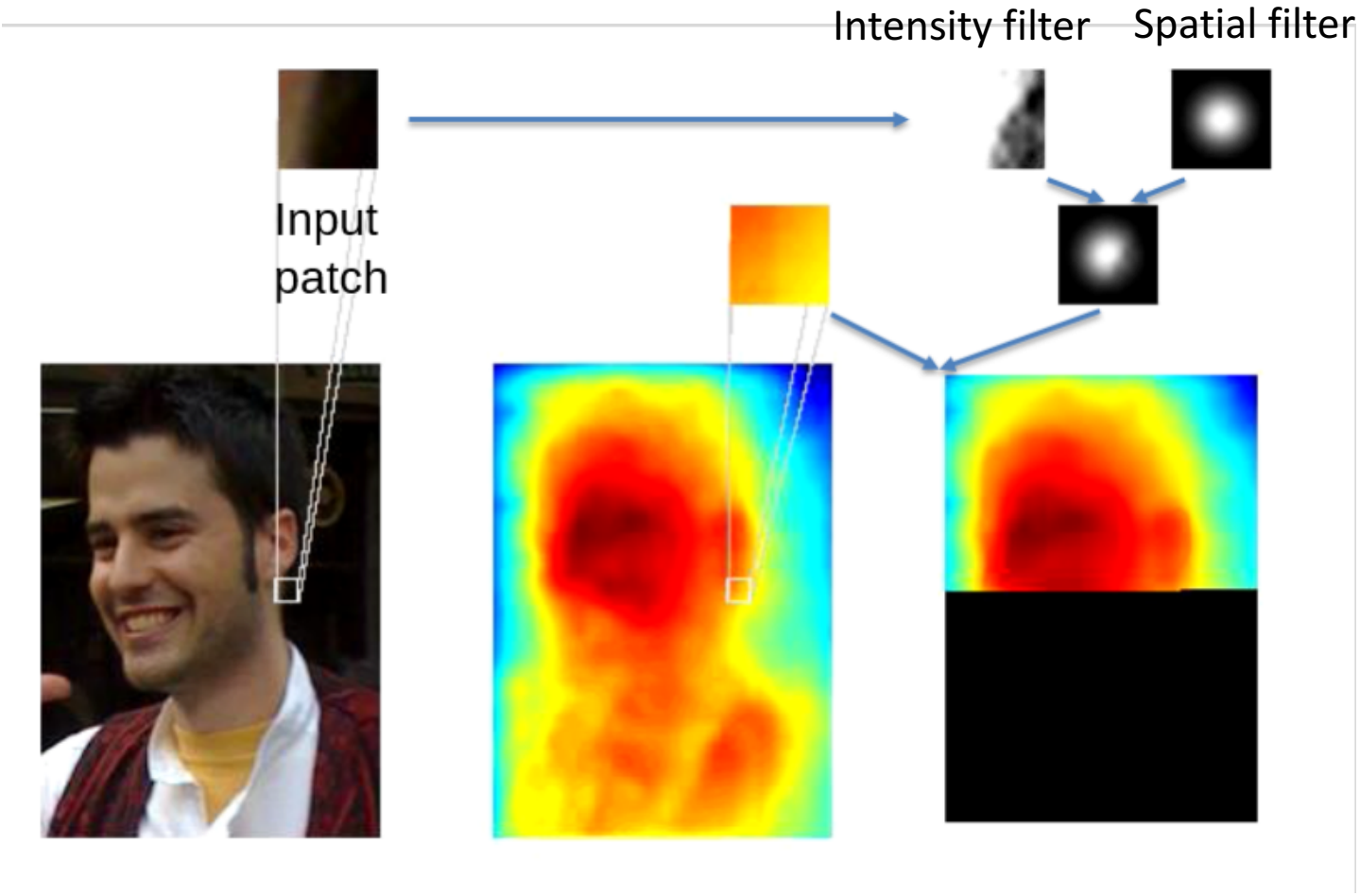
Related works to come:
Yang et al. CVPR 2009
He et al. ECCV 2010
Barron et al. ECCV 2016

Adams et al. Eurographics 2009.

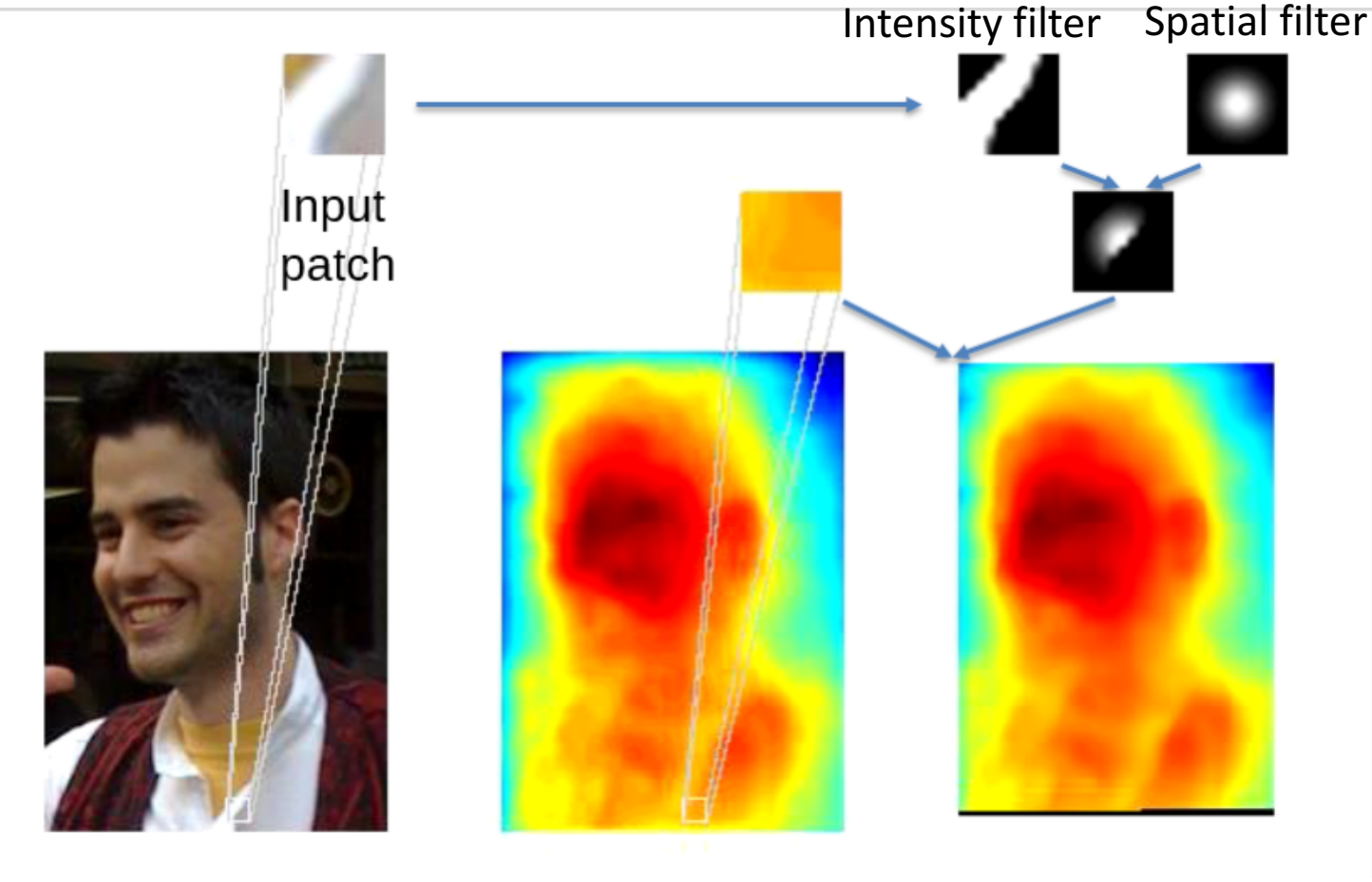
Bilateral filter



Bilateral filter



Bilateral filter

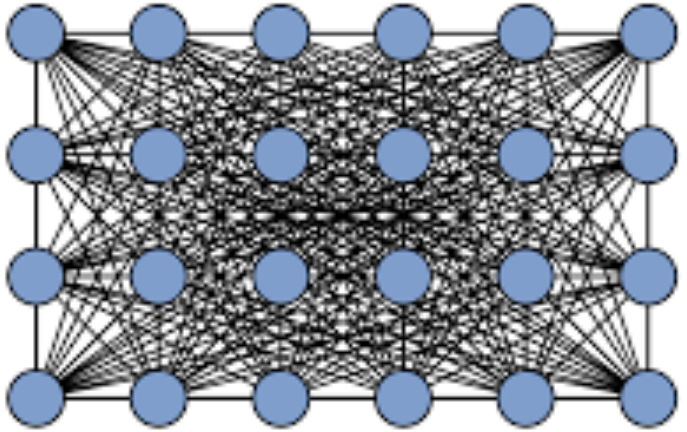


Fully-connected CRFs

- Pairwise energies are defined for every pixel pair in the image

$$E(\boldsymbol{X}|\boldsymbol{I}) = \sum_i unary(\boldsymbol{x}_i) + \sum_{i,j} pairwise(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

- Assuming Gaussian pairwise potentials
- Use approximate mean field inference



Mean field in fully connected CRFs

Algorithm 1 Mean field in fully connected CRFs

Initialize Q ▷ $Q_i(x_i) \leftarrow \frac{1}{Z_i}\exp\{-\phi_u(x_i)\}$

while not converged **do** ▷ See Section 6 for convergence analysis

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ **Message passing** from all X_j to all X_i

$\hat{Q}_i(x_i) \leftarrow \sum_{l \in \mathcal{L}} \mu^{(m)}(x_i, l) \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ **Compatibility transform**

$Q_i(x_i) \leftarrow \exp\{-\psi_u(x_i) - \hat{Q}_i(x_i)\}$ ▷ **Local update**

normalize $Q_i(x_i)$

end while

Krähenbühl et al. NIPS 2011.

Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\breve{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\breve{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while

Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

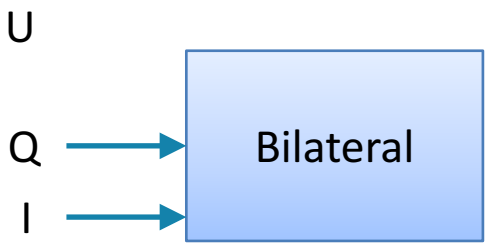
$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\check{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while



Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

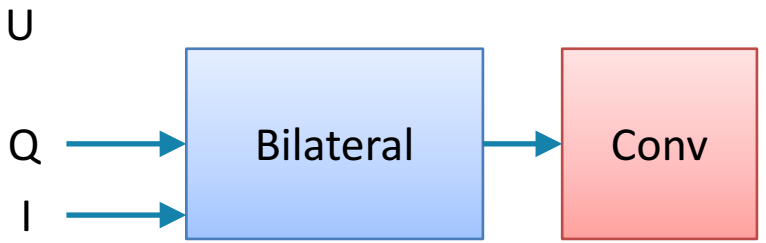
$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\check{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while



Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

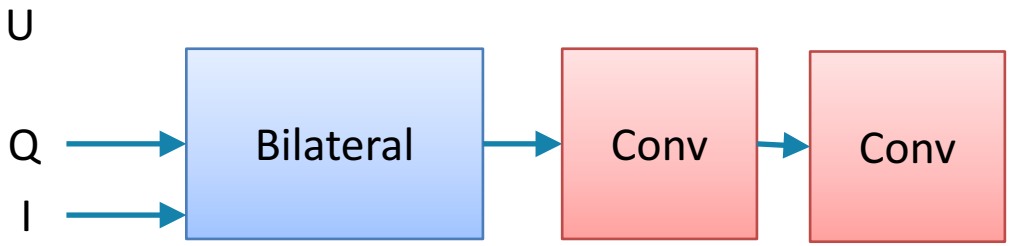
$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\check{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while



Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

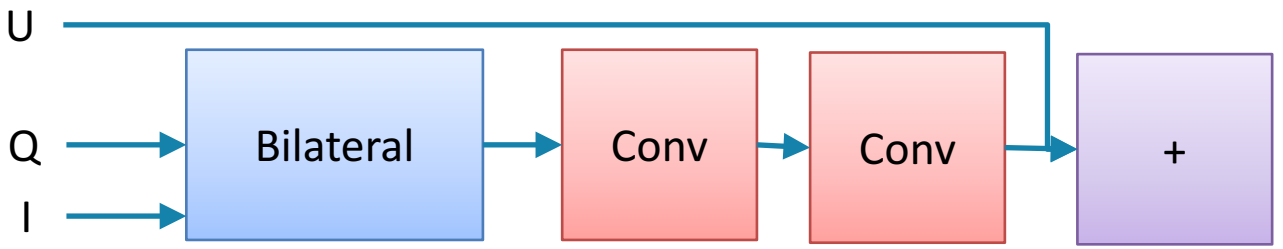
$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\check{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while



Conditional Random Fields as Recurrent Neural Networks

$Q_i(l) \leftarrow \frac{1}{Z_i(\mathbf{U})} \exp\left(U_i(l)\right)$ for all i ▷ Initialization

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ for all m ▷ Message Passing

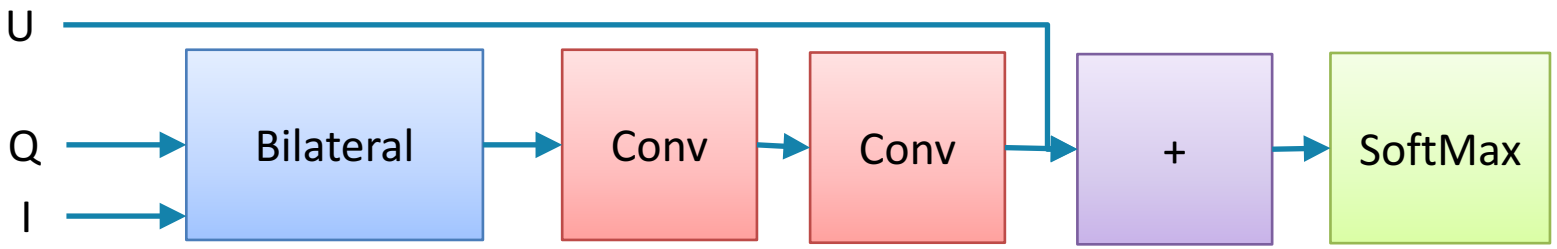
$\check{Q}_i(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_i^{(m)}(l)$ ▷ Weighting Filter Outputs

$\hat{Q}_i(l) \leftarrow \sum_{l' \in \mathcal{L}} \mu(l, l') \check{Q}_i(l')$ ▷ Compatibility Transform

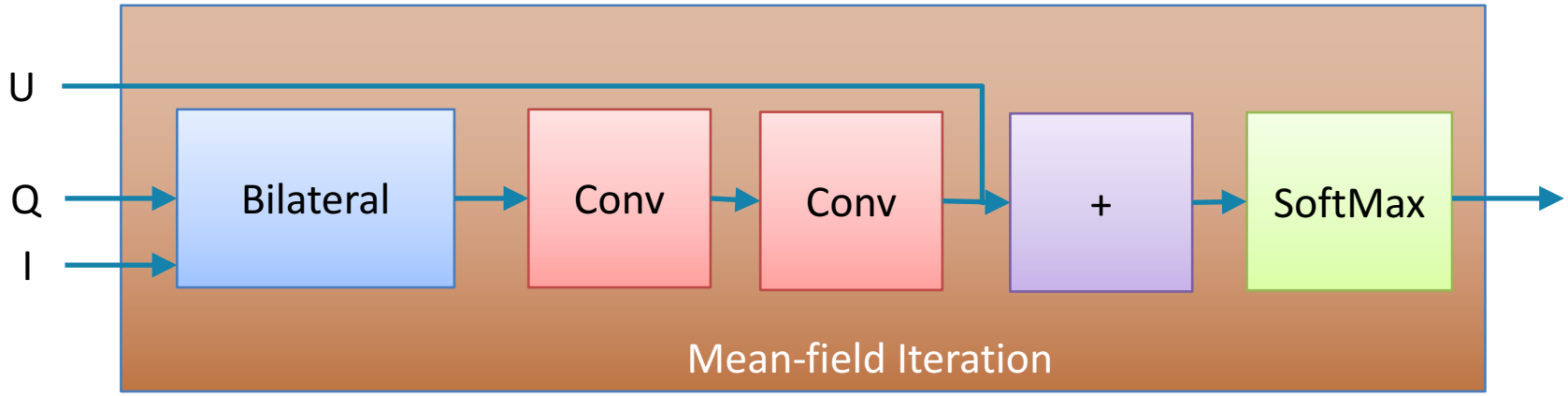
$\check{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$ ▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i(Q(\mathbf{X}))} \exp\left(\check{Q}_i(l)\right)$ ▷ Softmax Normalisation

end while

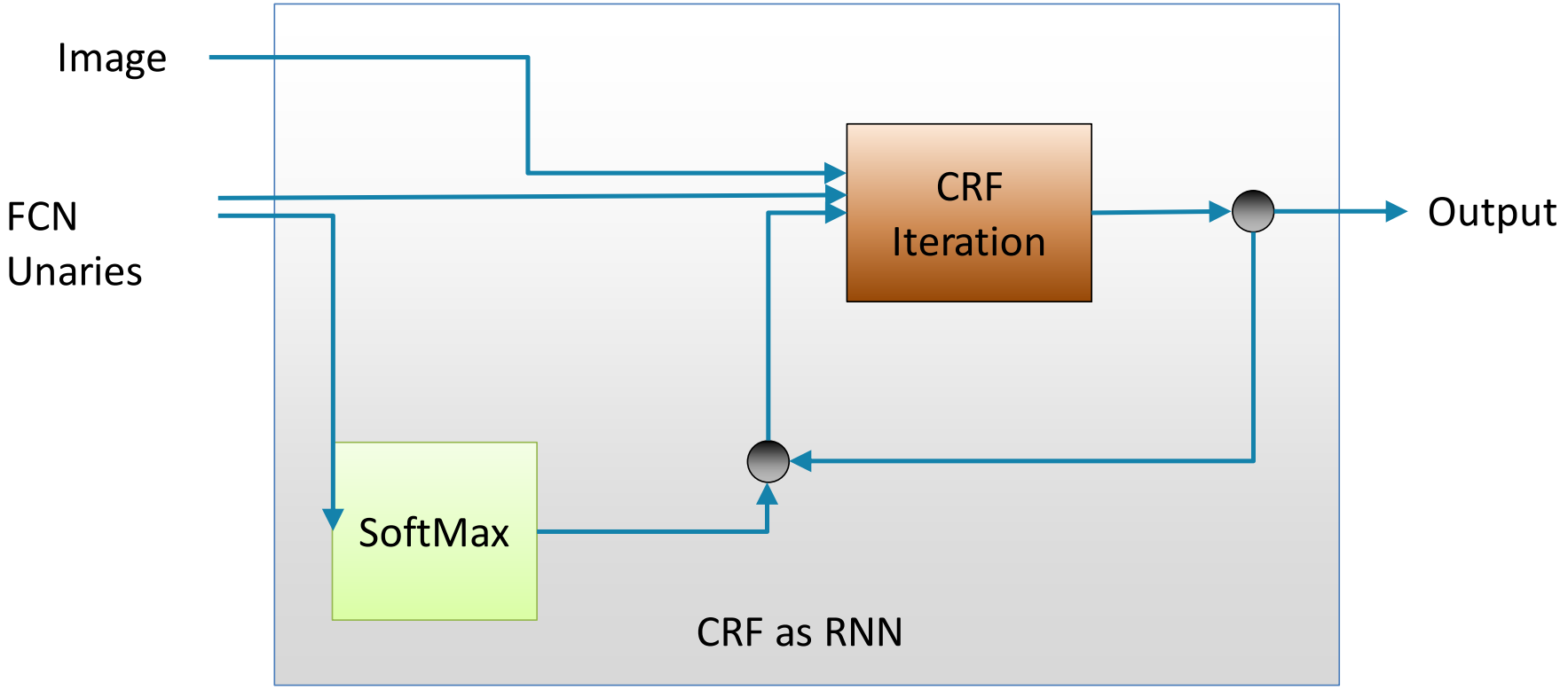


Conditional Random Fields as Recurrent Neural Networks



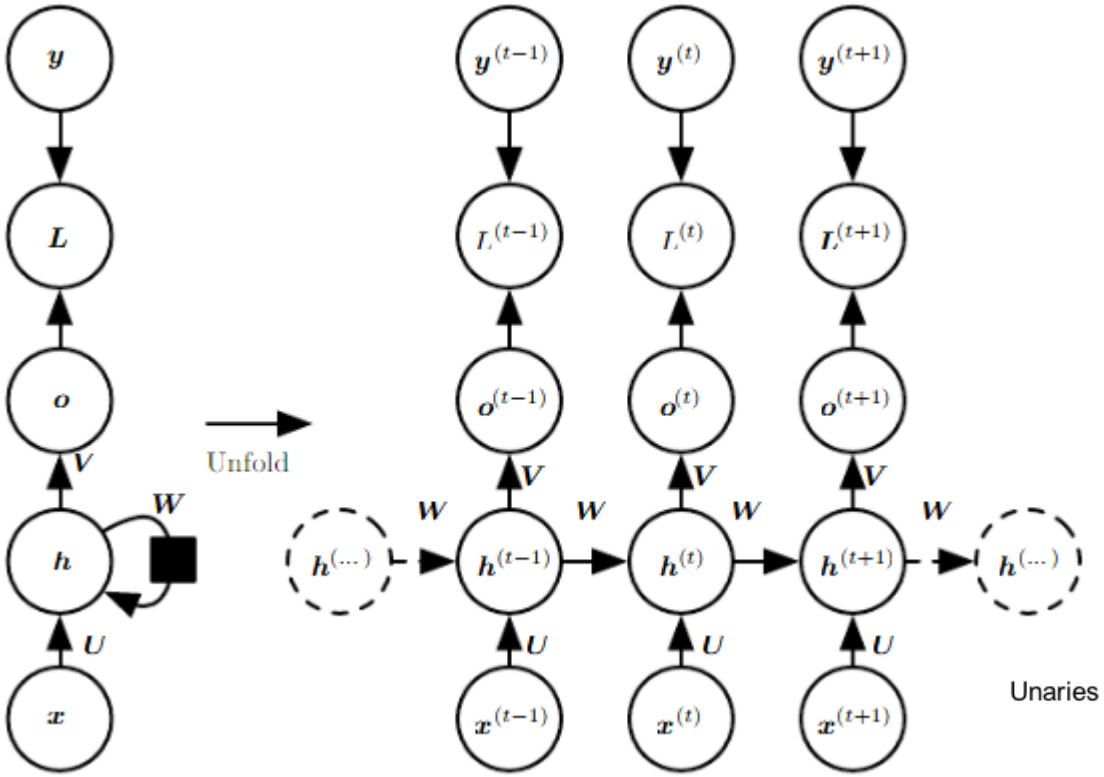
- Each of these blocks is differentiable → We can backprop

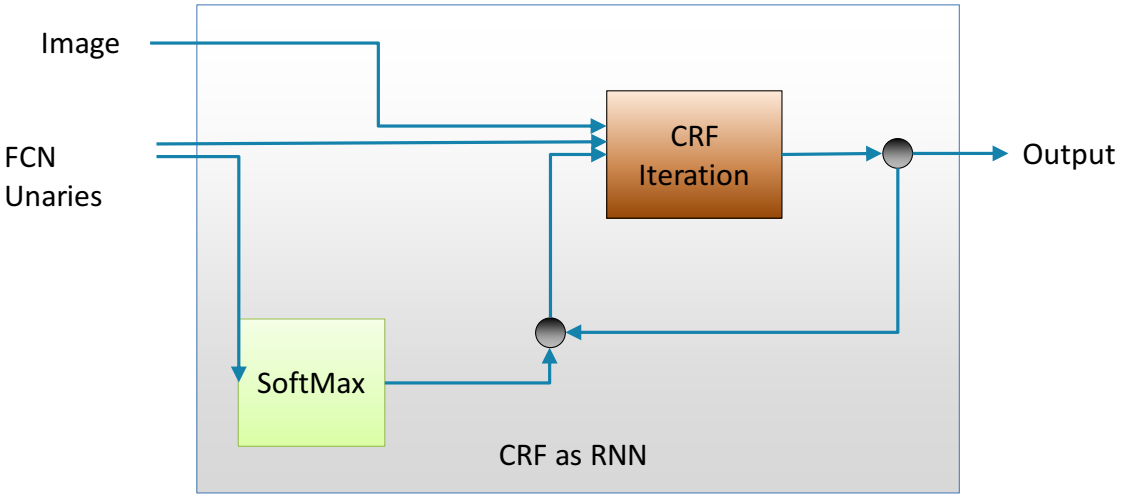
Conditional Random Fields as Recurrent Neural Networks

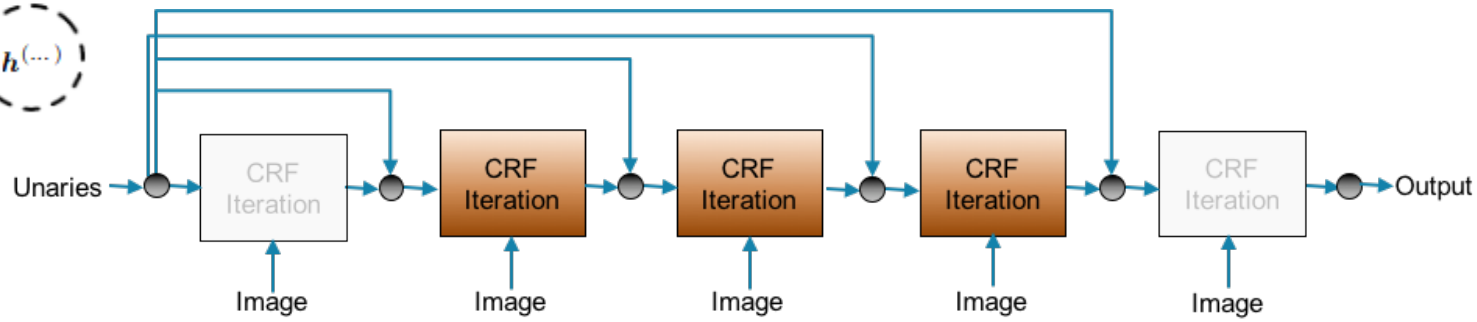


- Each of these blocks is differentiable → We can backprop

Conditional Random Fields as Recurrent Neural Networks

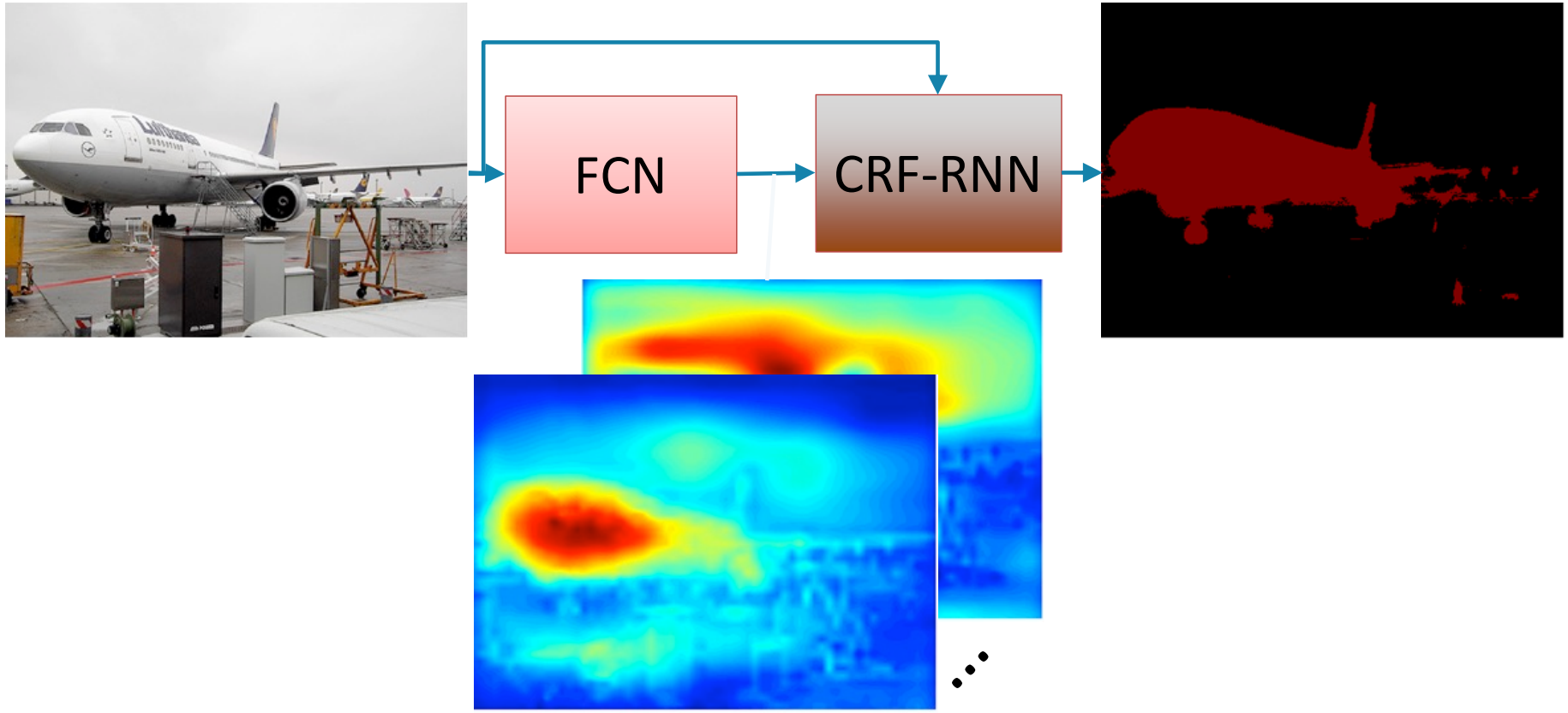






Unfolded CRF as RNN

CRF-RNN Layer in deep learning library

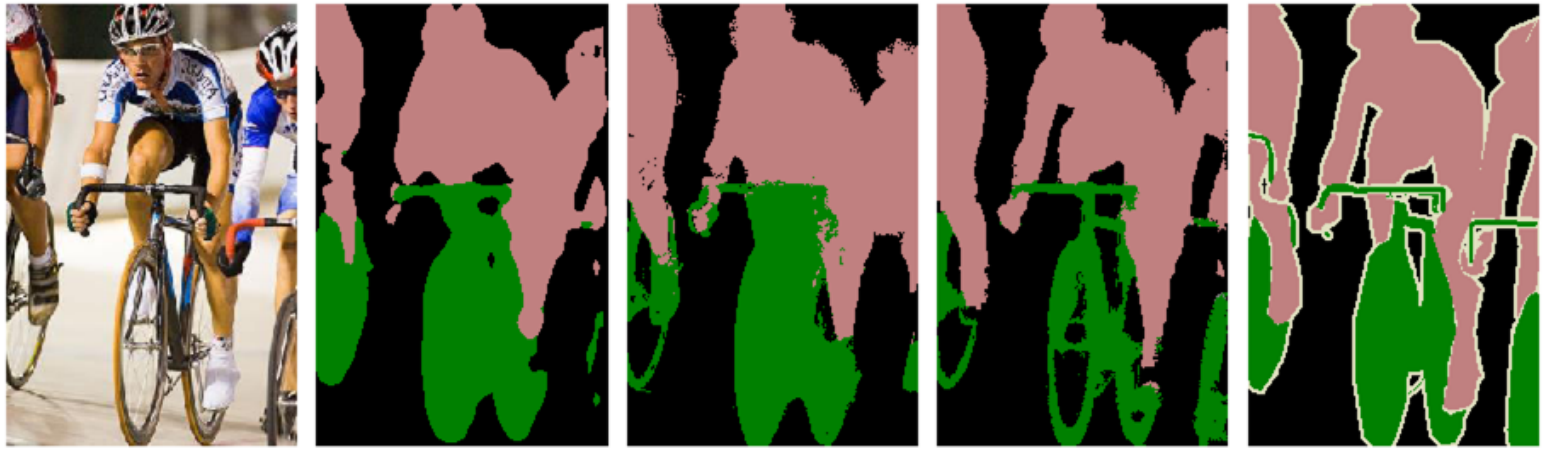


- Each of these blocks is differentiable → We can backprop. (Zheng et al. ICCV 2015)

https://github.com/torrvision/crfasrnn/

Experiments

	[Long et al, 2014]	[Chen et al, 2015]	Ours	Human Ground Truth
	FCN	FCN → CRF	FCN → CRF-RNN	
Mean IOU score	68.3	69.5	**72.9**	



Live Demo

Try your own image at:

http://crfasrnn.torr.vision

CRF as RNN

Semantic Image Segmentation Live Demo

Our work allows computer to recognize objects in images, what is distinctive about our work is that we also recover the 2D outline of the object.

Currently we have trained this model to recognize 20 classes. The demo below allows you to test our algorithm on your own images – have a try and see if you can fool it, if you get some good examples you can send them to us.

Why are we doing this? This work is part of a project to build augmented reality glasses for the partially sighted please read about it here smart-specs.

This demo is based on our paper *Conditional Random Fields as Recurrent Neural Networks*, which utilizes deep learning techniques and probabilistic graphical models for semantic image segmentation. [PDF] [Project] [Group]

Try your own image:

Provide an image URL | Process Image

Or upload it here:

Choose File | No file chosen

Share Tweet

Original image (hover to highlight segmented parts)

Semantic segmentation

Objects appearing in the image:

Bicycle | Person

Objects not appearing in the image:

Aeroplane	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow
Dining table	Dog	Horse	Motorbike	Potted plant	Sheep	Sofa	Train	TV/Monitor

Example: Recognizing cats

Cat

Example

Original image (hover to highlight segmented parts)

Semantic segmentation

Objects appearing in the image:

Bicycle | Person | Sheep

Not-so-good Example

Original image (hover to highlight segmented parts)

Semantic segmentation

Objects appearing in the image:

Bird Person

Tricky examples

Original image (hover to highlight segmented parts)

Semantic segmentation

Objects appearing in the image:

Bird | Dog | Person

http://crfasrnn.torr.vision

Image: http://i.huffpost.com/gen/1478236/thumbs/s-DIRD6-large640.jpg

Conclusion

- CNNs yield a coarse prediction on pixel-labeled tasks.
- CRFs improve the result by accounting for the contextual information in the image.
- Learning the whole pipeline end-to-end significantly improves the results.

Higher Order Conditional Random Fields in Deep Neural Networks

Failure Cases of CRF-as-RNN

- Poor unaries
  - Inference with pairwise potentials cannot help if unaries are poor

Failure Cases of CRF-as-RNN

- Poor unaries
  - Inference with pairwise potentials cannot help if unaries are poor
  - Cues from *object detectors* can help in this regard
  - Object detectors can “fire” over regions which have poor/incorrect unaries

[1] Wojek and Schiele. *ECCV* 2008. [2] Ladicky etal. *ECCV* 2010 [3] Yao etal. *CVPR 2012*

Failure Cases of CRF-as-RNN

- Poor unaries
  - Inference with pairwise potentials cannot help if unaries are poor
  - Cues from *object detectors* can help in this regard
  - Object detectors can “fire” over regions which have poor/incorrect unaries
- Spurious noise in segmentations
  - Pairwise consistency not enough

Failure Cases of CRF-as-RNN

- Poor unaries
  - Inference with pairwise potentials cannot help if unaries are poor
  - Cues from *object detectors* can help in this regard
  - Object detectors can “fire” over regions which have poor/incorrect unaries
- Spurious noise in segmentations
  - Pairwise consistency not enough
  - Enforce consistency over regions
  - *Superpixel based potentials* can help here

[1] Kohli etal. *IJCV 2009* [2] Ladicky etal. *ICCV 2009* [3] Vineet etal. *ECCV 2012*

Conditional Random Fields

$$P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{I}) = \frac{1}{Z(I)} \exp[-E(\boldsymbol{x} \mid \boldsymbol{I})]$$

$$E(\boldsymbol{x}) = \sum_{c \in C} \psi_c(\boldsymbol{x_c})$$

In our case

$$E(x) = \underbrace{\sum_i \psi_i^U(x_i)}_{\text{Unaries from CNN}} + \underbrace{\sum_{i<j} \psi_{i,j}^P(x_i, x_j)}_{\text{Pairwise [1]}} + \underbrace{\sum_d \psi_d^{Det}(\boldsymbol{x}_d)}_{\text{Detection potentials}} + \underbrace{\sum_s \psi_s^{SP}(\boldsymbol{x}_s)}_{\text{Superpixel potentials}}$$

[1] Krahenbuhl and Koltun. *NIPS* 2011.

Mean Field

- An approximate method. Works well in practice

```
Initialise
```

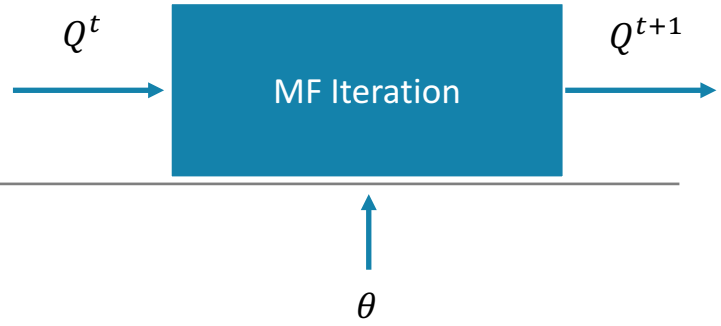
$$Q_i = \frac{1}{Z_i} \exp(U_i(l))$$

```
while not converged do
```

$$Q^{t+1}(V_i = l) = \frac{1}{Z_i} \exp\left(- \sum_{c \in C} \sum_{\{\boldsymbol{v_c} \mid v_i = l\}} Q^t(\boldsymbol{v_{c-i}}) \psi(\boldsymbol{v}_c) \right)$$

```
end
```

Mean Field



- An approximate method. Works well in practice

`Initialise`

$$Q_i = \frac{1}{Z_i}\exp(U_i(l))$$

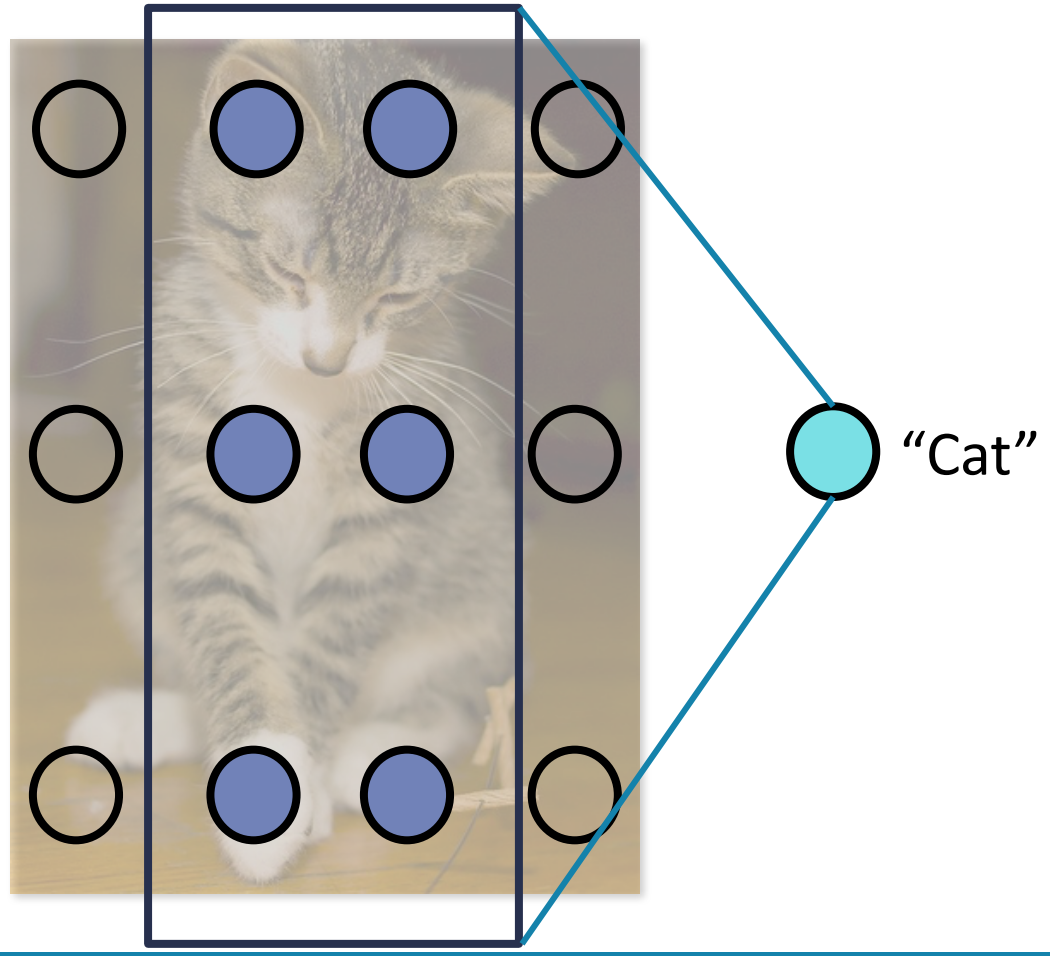
`while not converged do`

$$Q^{t+1}(V_i = l) = \frac{1}{Z_i}\exp\left(-\sum_{c \in C}\sum_{\{\boldsymbol{v}_c \mid v_i = l\}} Q^t(\boldsymbol{v_{c-i}})\psi(\boldsymbol{v}_c\,;\theta)\right)$$

Linear with respect to Q

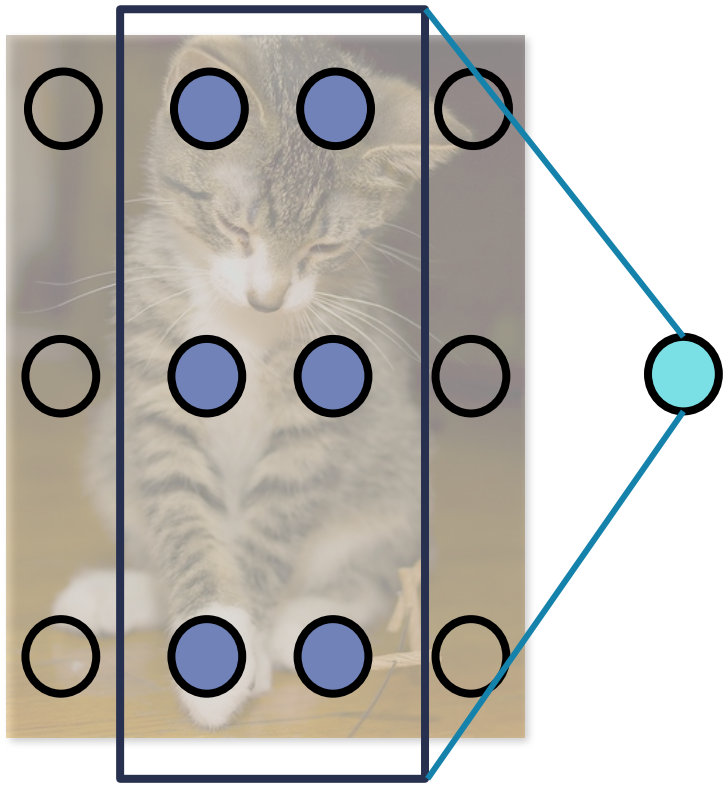
`end`

Detection Potential



Detection Potential

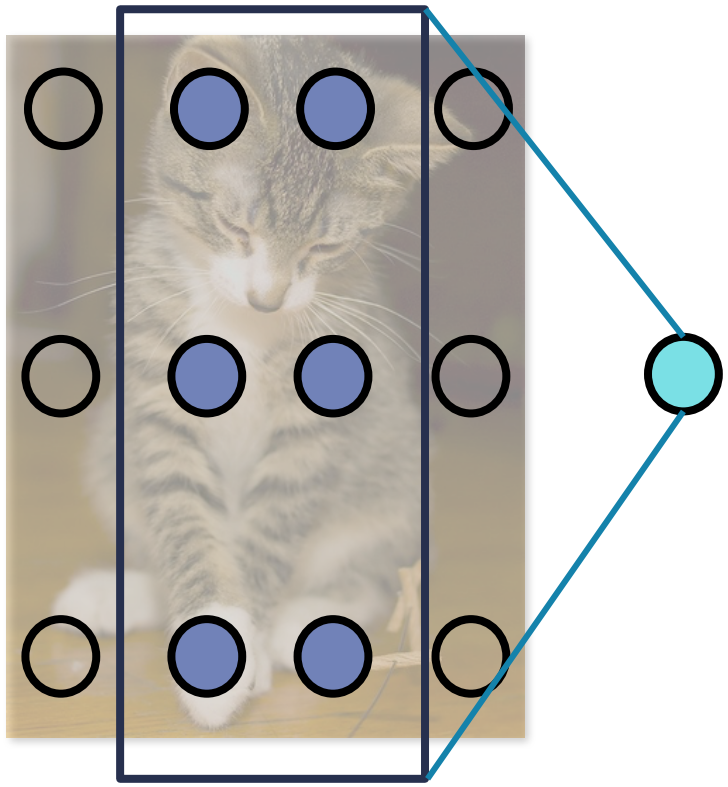
- Assume we have D object detections for a given image
- d^{th} detection is of the form (l_d, s_d, F_d, B_d)
  - $l_d \in \mathcal{L}$ is the class label of the detection
  - $s_d \in [0,1]$ is the detection score
  - $F_d \subseteq \{1,2 \ \ldots N\}$ is the set of pixels belonging to the foreground of the detection. Obtained by foreground/background segmentation (ie GrabCut [1]) on the detection bounding box
  - B_d are the set of pixels falling inside the bounding box (use this later)



[1] Rother et al. *TOG* 2004.

Detection Potential

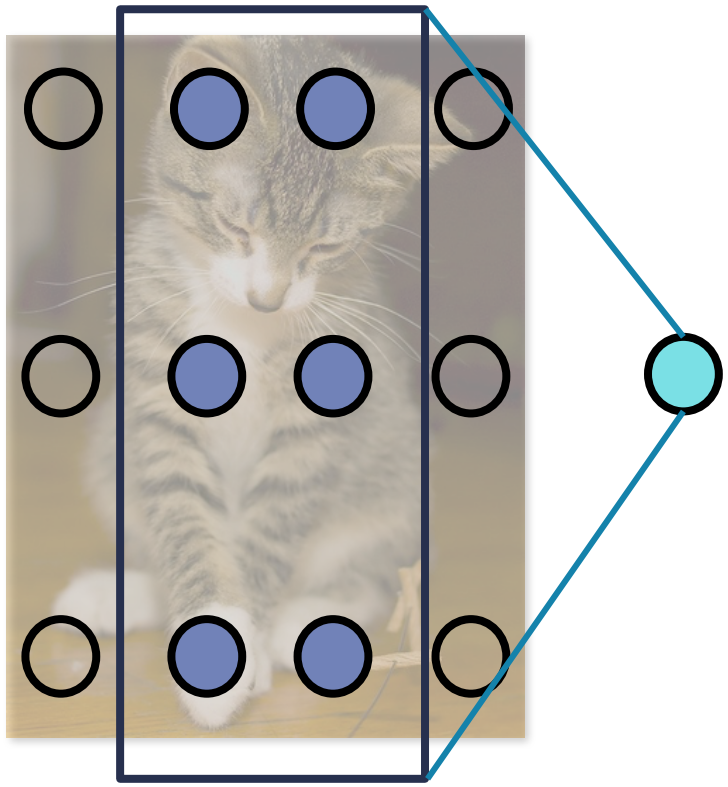
- Assume we have D object detections for a given image
- d^{th} detection is of the form (l_d, s_d, F_d)
- Introduce binary latent variables, $Y_1, Y_2 \ldots Y_D$ - one for each detection
  - Models whether the detection hypothesis has been accepted or not
  - $Y_d = 1$ after inference indicates that the detection hypothesis has been accepted
  - $Y_d = 1$ Initialised to s_d, the score of the object detector. $Y_d = 0$ Initialised to $1 - s_d$



Detection Potential

- Assume we have D object detections for a given image
- d^{th} detection is of the form (l_d, s_d, F_d)
- Introduce binary latent variables, $Y_1, Y_2 \ldots Y_D$ - one for each detection

$$\psi_d^{Det}(\boldsymbol{X_d} = \boldsymbol{x_d}, Y_d = y_d) = \begin{cases} w_{det}(l_d)\frac{s_d}{n_d}\sum_{i=1}^{n_d}\left[x_d^{(i)} = l_d\right] & if\, y_d = 0 \\ w_{det}(l_d)\frac{s_d}{n_d}\sum_{i=1}^{n_d}\left[x_d^{(i)} \neq l_d\right] & if\, y_d = 1 \end{cases}$$



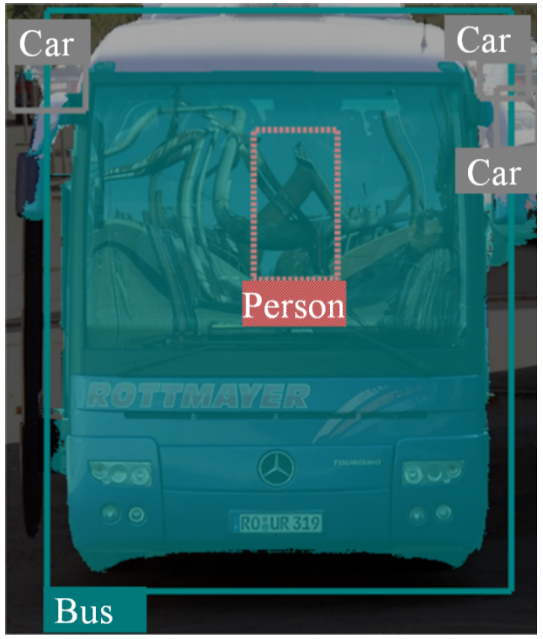
Detection Potential – Foreground Example

Method robust to poor foreground segmentations since detection potentials are another cue which must be taken in context of other energies in the CRF.

Detection Potential – Examples

CRF-as-RNN

Ours



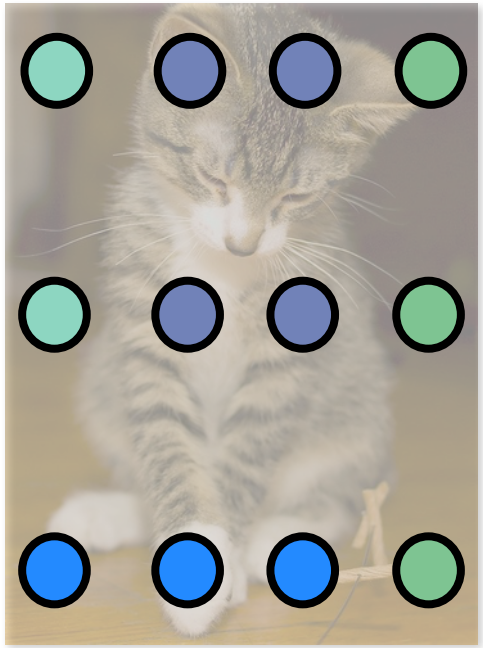
CRF-as-RNN

Ours

Detection potentials can help overcome cases where our unaries are poor

Superpixel Potential

- Enforce consistency over entire regions obtained by superpixels.
- P^n-Potts model type energy [1-3]
- Low cost if all the random variables within a superpixel are assigned the same label. High cost otherwise
- Use superpixels over multiple scales, which do not necessarily have to form a hierarchy
- Reduces spurious noise in segmentations
- $$\psi_s^{SP}(\boldsymbol{X_s} = \boldsymbol{x_s}) = \begin{cases} w_{low}(l) & \text{if all } x_s^{(i)} = l \\ w_{high} & \text{otherwise} \end{cases}$$



Each colour represents a different superpixel

[1] Kohli et al. *CVPR* 2007. [2] Ladicky et al. *ICCV* 2009 [3] Vineet et al. *IJCV* 2014

Superpixel Potential - Example

CRF-as-RNN

One “layer” of superpixels

Output using superpixel potentials

Mean Field Updates

Initialise

$$Q_i = \frac{1}{Z_i}\exp(U_i(l))$$

while not converged do

$$Q^{t+1}(V_i = l) = \frac{1}{Z_i}\exp\left(-\sum_{c \in C}\sum_{\{\boldsymbol{v}_c \mid v_i = l\}} Q^t(\boldsymbol{v_{c-i}})\psi(\boldsymbol{v}_c)\right)$$

end

- Need to compute mean field updates during inference
- What we need to implement in our neural network

$$Q^{t+1}(V_i = l) = \frac{1}{Z_i}\exp\left(-\sum_{c \in C}\sum_{\{\boldsymbol{v}_c \mid v_i = l\}} Q^t(\boldsymbol{v_{c-i}})\psi(\boldsymbol{v}_c)\right)$$

For superpixel potential:

$$\sum_{\{\boldsymbol{x}_s \mid x_s^{(i)} = l\}} Q(\boldsymbol{x}_{s\,-i})\psi_s^{SP}(\boldsymbol{x_s}) = w_{low}\prod_{j \in s, j \neq i} Q(X_j = l) + w_{high}\left(1 - \prod_{j \in s, j \neq i} Q(X_j = l)\right)$$

[1] Vineet et al. IJCV 2014

Mean Field Updates

Initialise

$$Q_i = \frac{1}{Z_i}\exp(U_i(l))$$

while not converged do

$$Q^{t+1}(V_i = l) = \frac{1}{Z_i}\exp\left(-\sum_{c\in C}\sum_{\{\boldsymbol{v_c}\mid v_i=l\}} Q^t(\boldsymbol{v_{c-i}})\psi(\boldsymbol{v_c})\right)$$

end

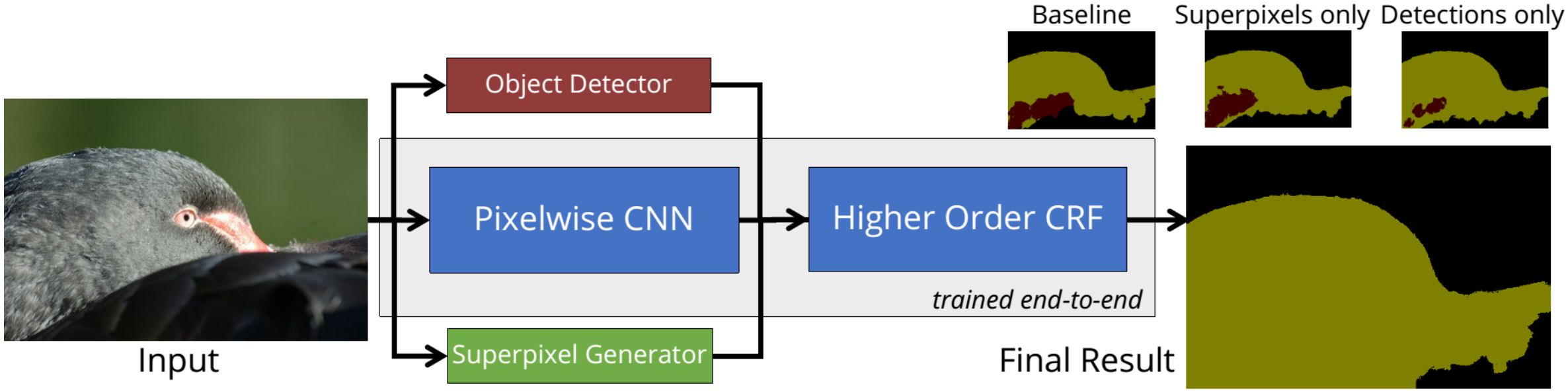
- Need to compute mean field updates during inference
- What we implement in our neural network

For detection potential:

$$\sum_{\{(\boldsymbol{x}_d,y_d)|x_d^{(i)}=l\}} Q(\boldsymbol{x}_{d\,-i}, y_d)\psi_d^{Det}(\boldsymbol{x}_d, y_d) = \begin{cases} w_{Det}(l_d)\,\frac{s_d}{n_d}Q(Y_d=0) & \text{if } l = l_d \\ w_{Det}(l_d)\,\frac{s_d}{n_d}Q(Y_d=1) & \text{otherwise} \end{cases}$$

$$\sum_{\{(\boldsymbol{x}_d,y_d)|y_d=b\}} Q(\boldsymbol{x}_d)\psi_d^{Det}(\boldsymbol{x}_d, y_d) = \begin{cases} w_{Det}(l_d)\frac{s_d}{n_d}\sum_{i=1}^{n_d} Q\left(X_d^{(i)} = l_d\right) & \text{if } b = 0 \\ w_{Det}(l_d)\frac{s_d}{n_d}\sum_{i=1}^{n_d}\left(1 - Q\left(X_d^{(i)} = l_d\right)\right) & \text{if } b = 1 \end{cases}$$

Putting it together

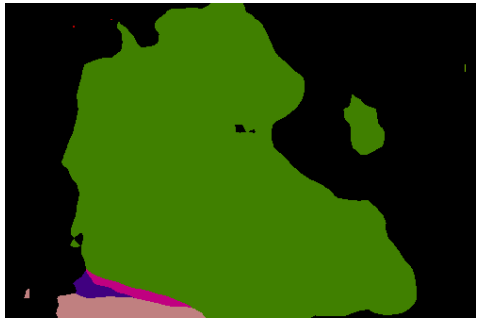


[1] Ren et al. *NIPS* 2015. [2] Felzenszwalb and Huttenlocher. *IJCV* 2004.

Results

FCN

$$E(\boldsymbol{x}) = \sum_i \psi_i^U(x_i)$$

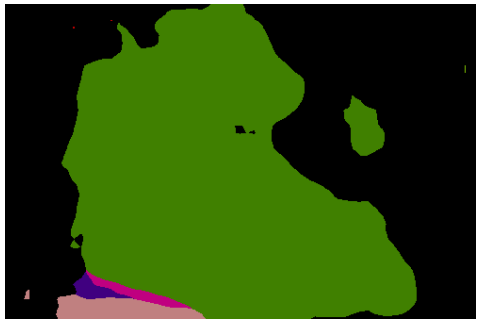


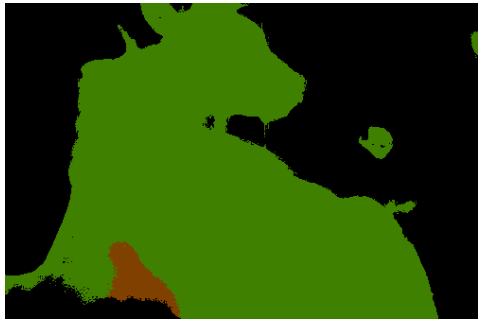
Results

Pairwise

$$E(\boldsymbol{x}) = \sum_i \psi_i^U(x_i) + \sum_{i<j} \psi_{i,j}^P(x_i, x_j)$$

FCN



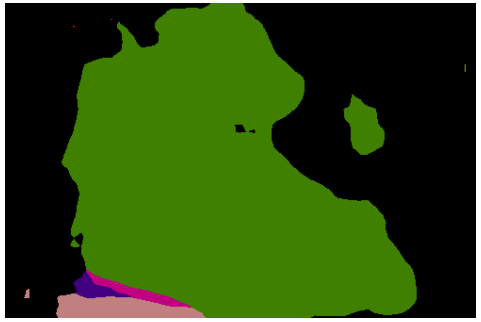


Results

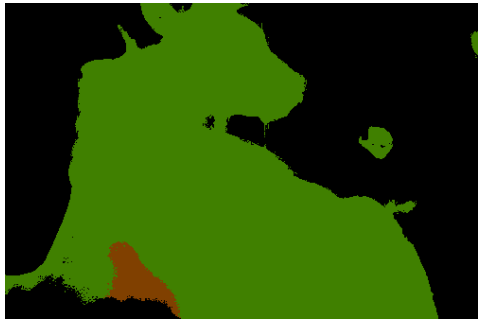
Superpixels

$$E(\boldsymbol{x}) = \sum_i \psi_i^U(x_i) + \sum_{i<j} \psi_{i,j}^P(x_i, x_j) + \sum_s \psi_s^{SP}(\boldsymbol{x}_s)$$

FCN



Pairwise

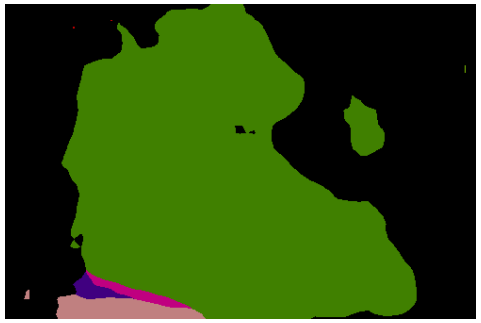


Results

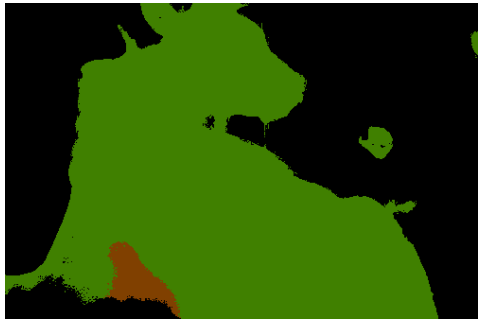
Detections

$$E(\boldsymbol{x}) = \sum_i \psi_i^U(x_i) + \sum_{i<j} \psi_{i,j}^P(x_i, x_j) + \sum_d \psi_d^{Det}(\boldsymbol{x}_d, y_d)$$

FCN



Pairwise

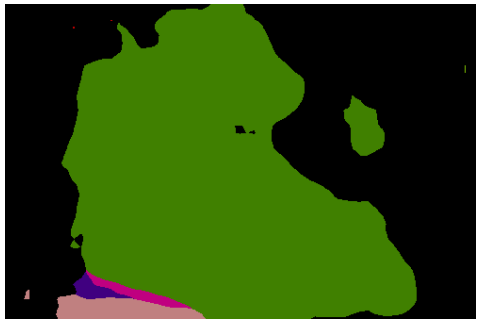


Superpixels

Results

$$E(\boldsymbol{x}) = \sum_i \psi_i^U(x_i) + \sum_{i<j} \psi_{i,j}^P(x_i, x_j) + \sum_s \psi_s^{SP}(\boldsymbol{x}_s) + \sum_d \psi_d^{Det}(\boldsymbol{x}_d, y_d)$$

FCN



Pairwise

Superpixels

Detections

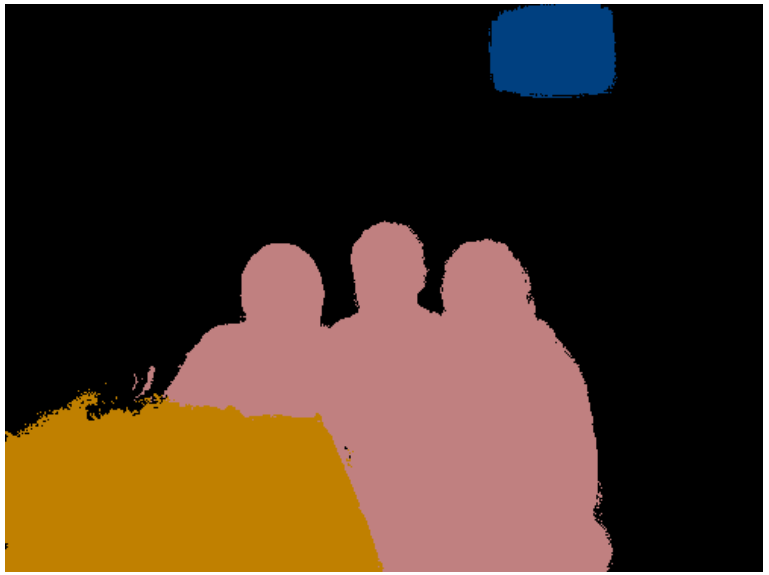
All

Results

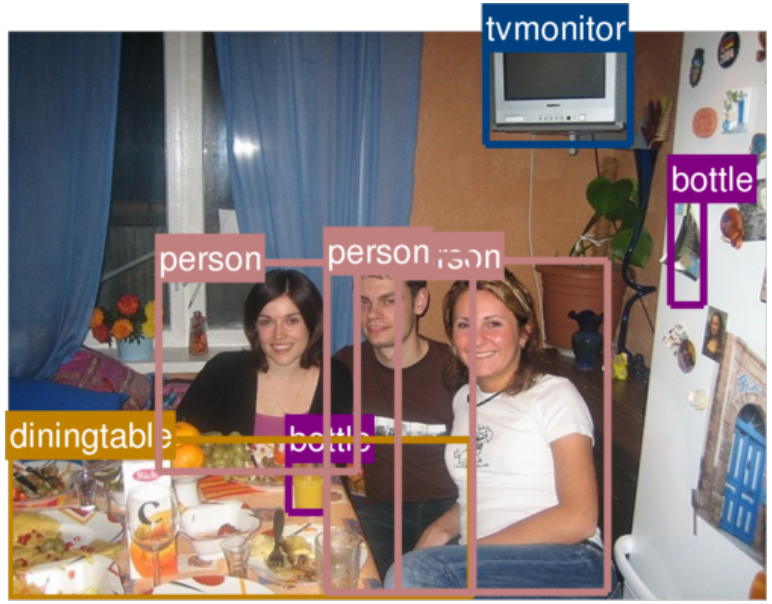
- On PASCAL VOC 2012 reduced validation set

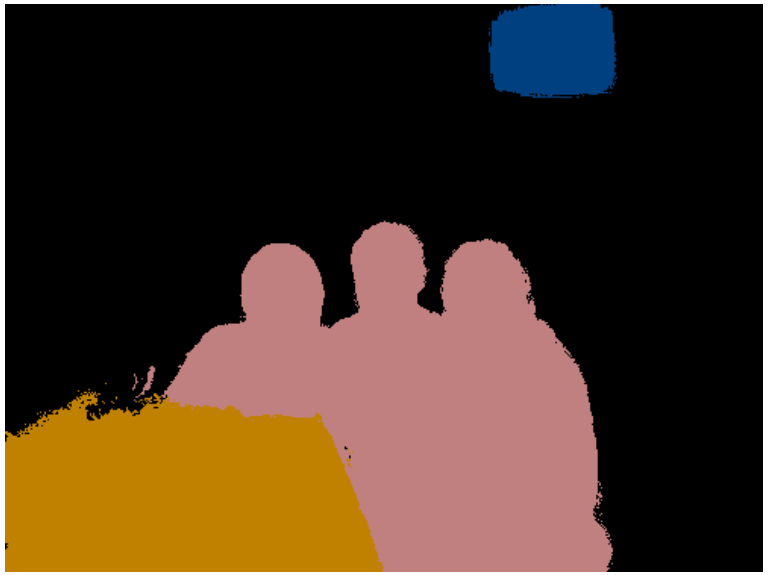
Method	Mean IoU [%]
FCN (Unary only)	68.3
Pairwise	72.9
Superpixels	74.0
Detections	74.9
Superpixels and Detections	75.8

Instance Segmentation



Embarrassingly Simple Instances





Input

End-to-end object segmentation subnetwork

Pixelwise CNN

Higher Order CRF

Object Detector

$K+1$

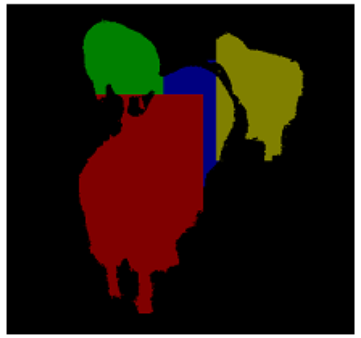
$D+1$

Instance Identification

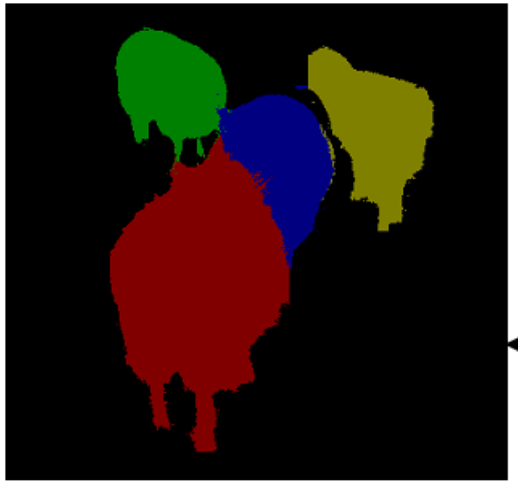
Instance CRF

End-to-end Instance segmentation subnetwork

Object Segmentation



Identified Instances



Output

Instance CRF

- X – Random variable, for each pixel, denoting semantic class. Takes on one of $K + 1$ labels
- Y – Random variable, for every detection output. Total of D detections
- V – Random variable, for each pixel, denoting instance. Takes on one of $D + 1$ labels

$$\Pr(v_i = k) = \begin{cases} \frac{1}{Z(\boldsymbol{Y}, \boldsymbol{Q})} Q_i(l_k) \Pr(Y_k = 1) & \text{if } i \in B_k \\ 0 & \text{otherwise} \end{cases}$$

$$E(\boldsymbol{v}) = \sum_i \psi_i^U(v_i) + \sum_{i<j} \psi_{i,j}(v_i, v_j) \qquad \psi_i^U(v_i) = -\ln \Pr(v_i)$$

Conclusion

- Higher Order potentials are effective in deep neural networks
- Higher Order potentials can also be trained end-to-end
- Allows a simple method of performing instance segmentation

Recurrent Instance Segmentation

Quick experiment

- Have a look at the following image

Quick experiment

- Have a look at the following image

Quick experiment

- What is in there?

Quick experiment

- What is in there?
  - Bottles
  - Question related to object recognition

Quick experiment

- What is in there?
  - Bottles
  - Question related to object recognition
- Where are they within the image?

Quick experiment

- What is in there?
  - Bottles
  - Question related to object recognition
- Where are they within the image?
  - About there
  - Question related to semantic segmentation

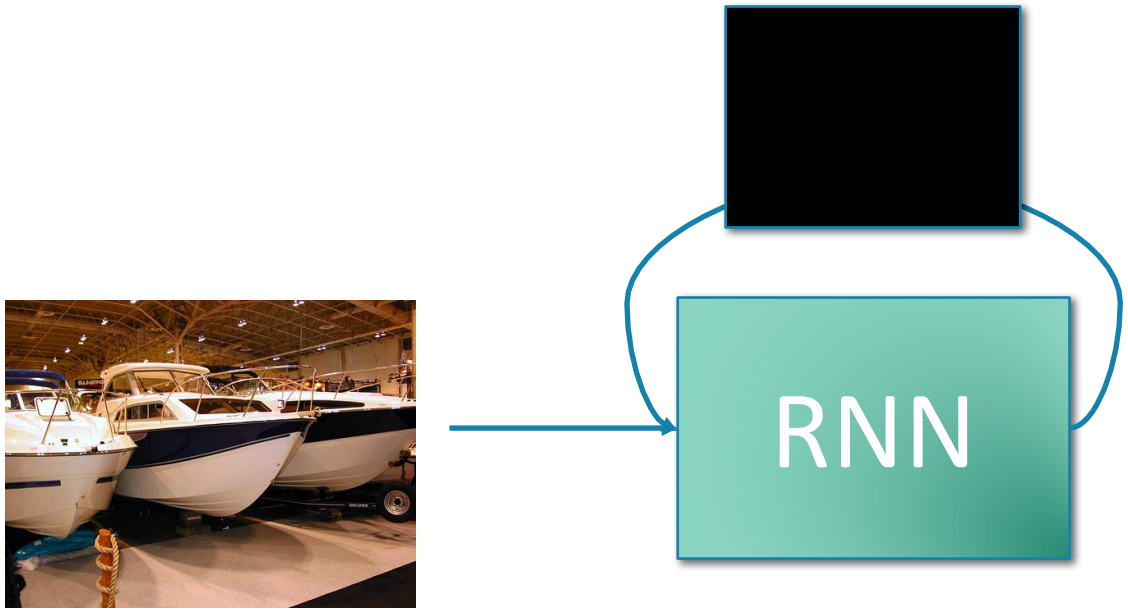
Quick experiment

- What is in there?
  - Bottles
  - Question related to object recognition
- Where are they within the image?
  - About there
  - Question related to semantic segmentation
- How many of them are there?

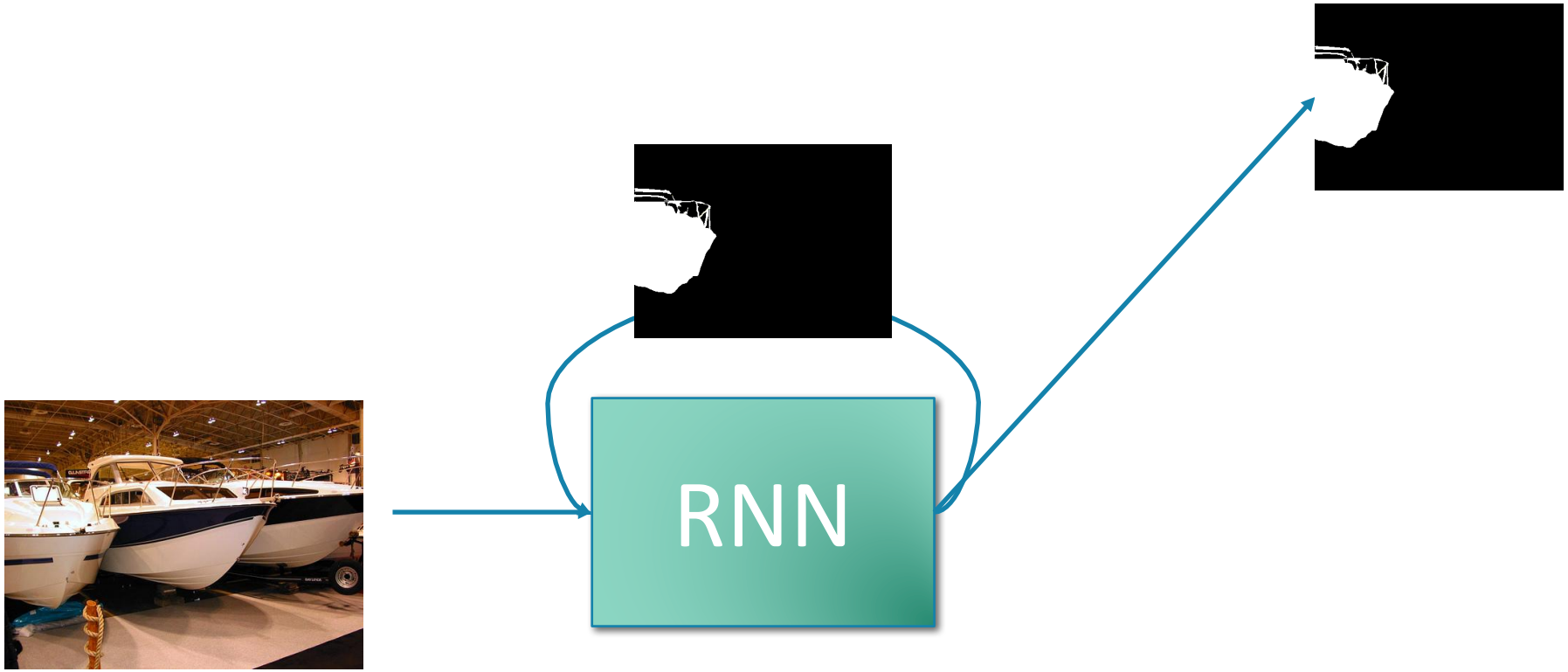
Quick experiment

- What is in there?
  - Bottles
  - Question related to object recognition
- Where are they within the image?
  - About there
  - Question related to semantic segmentation
- How many of them are there?
  - Question related to instance segmentation

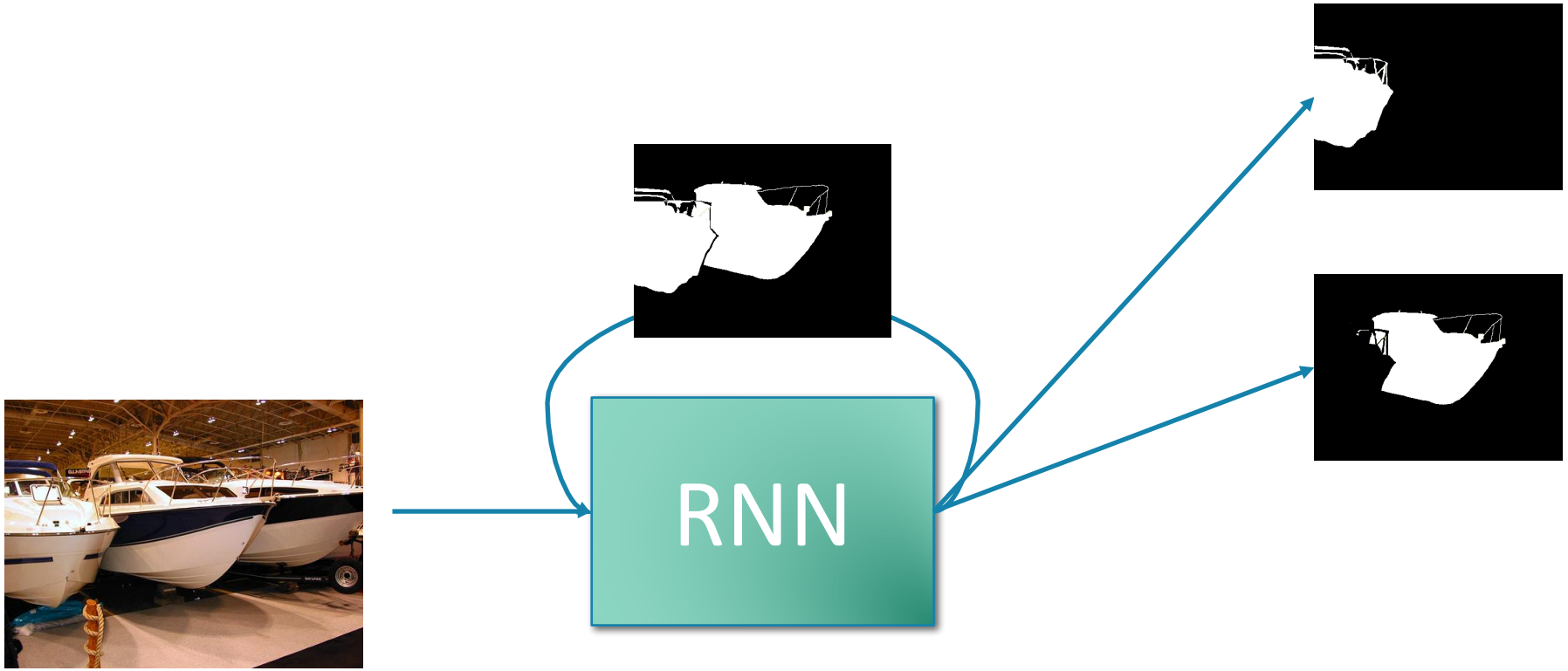
Recurrent Instance Segmentation



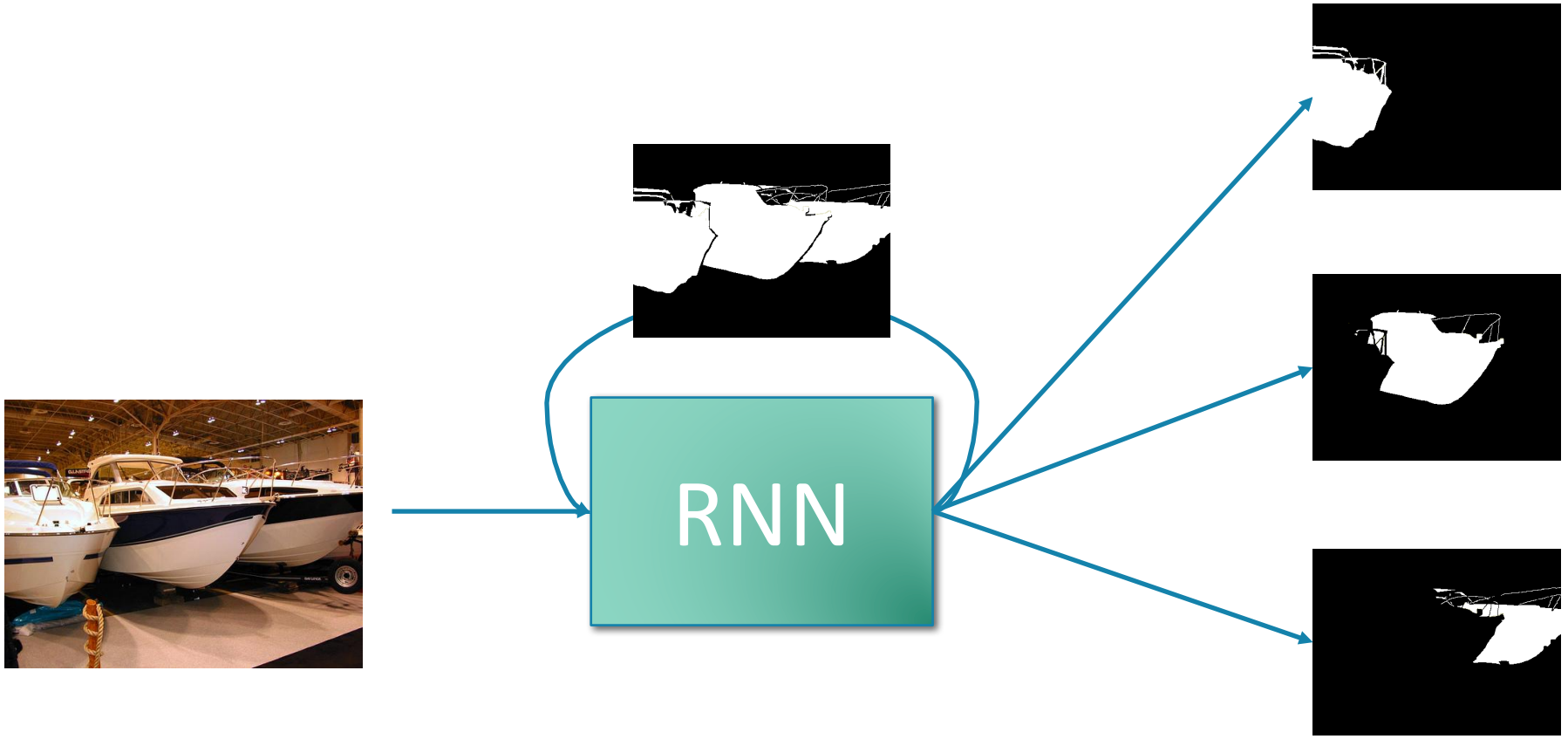
Recurrent Instance Segmentation



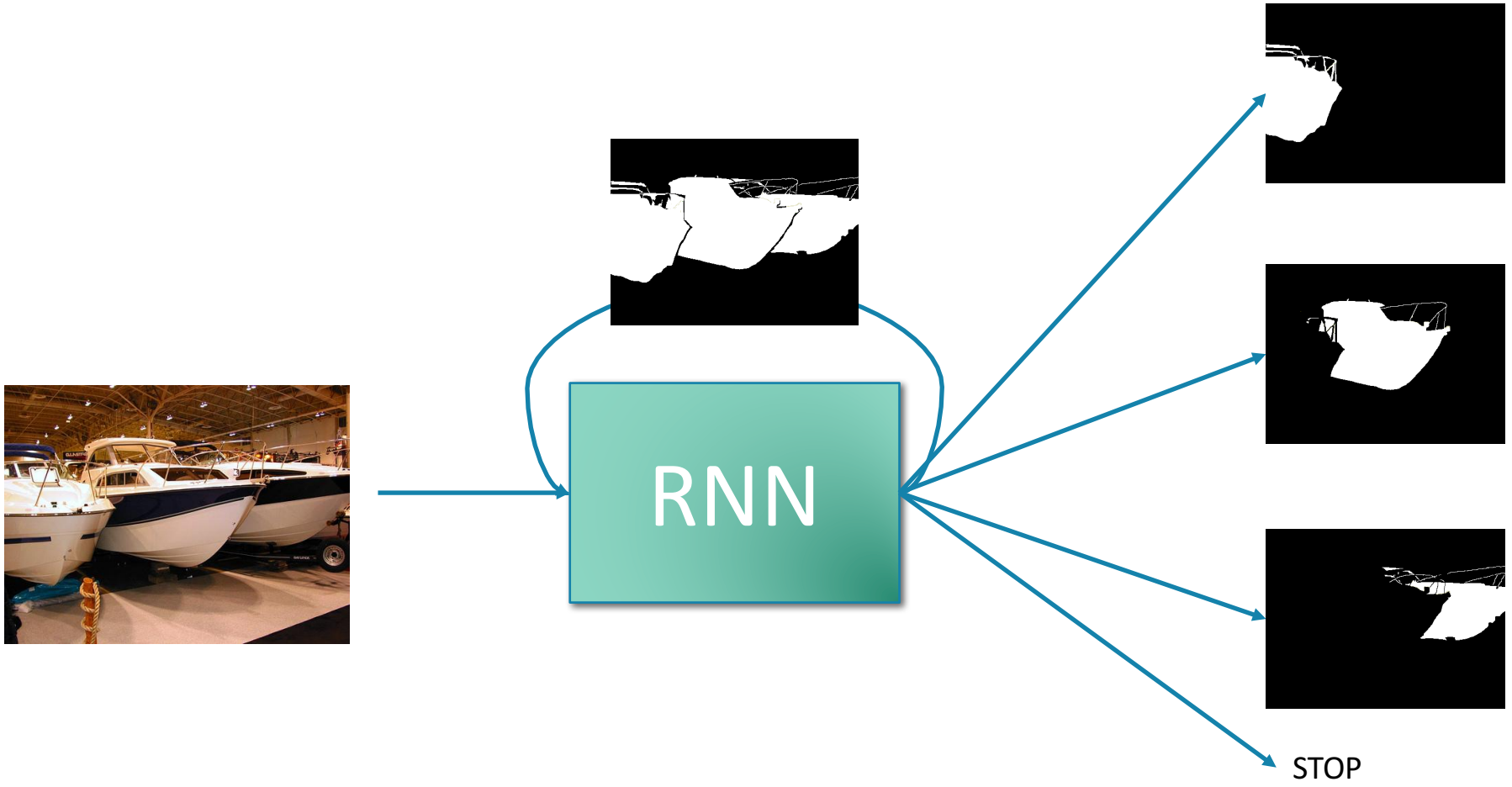
Recurrent Instance Segmentation



Recurrent Instance Segmentation

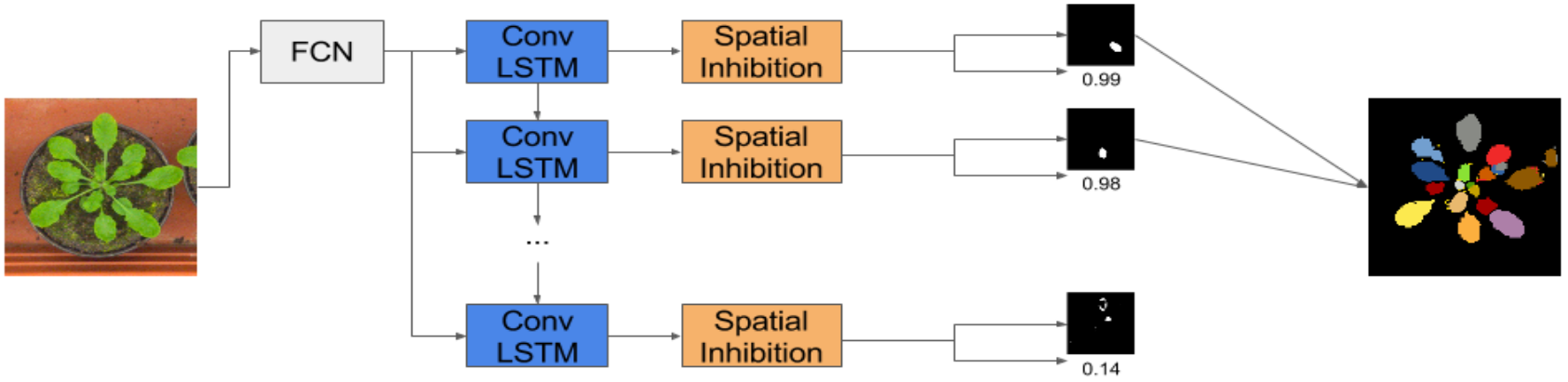


Recurrent Instance Segmentation



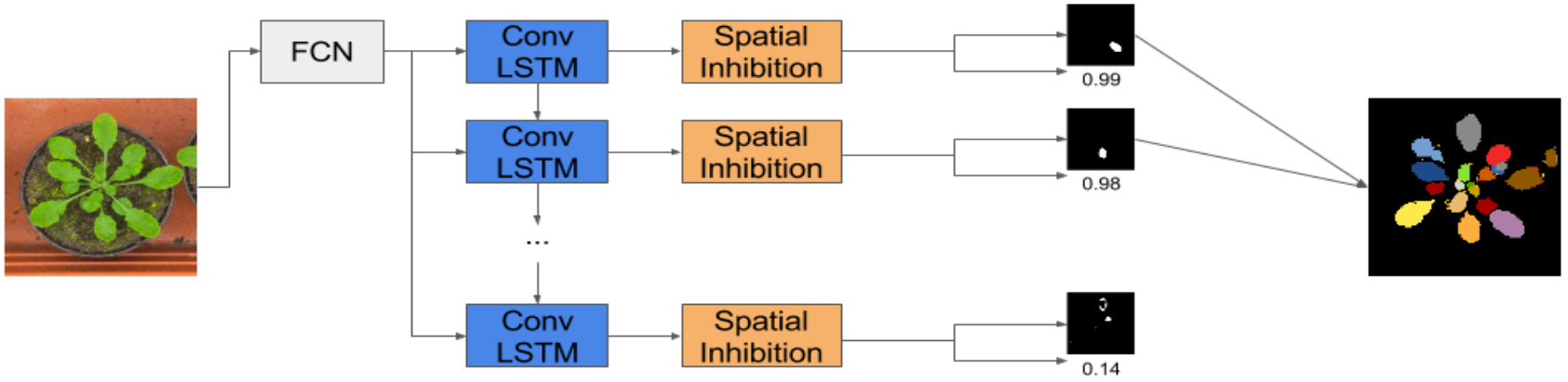
Recurrent Instance Segmentation

- Elements
  - Fully Convolutional Network
  - ConvLSTM
  - Spatial Inhibition
  - Loss function

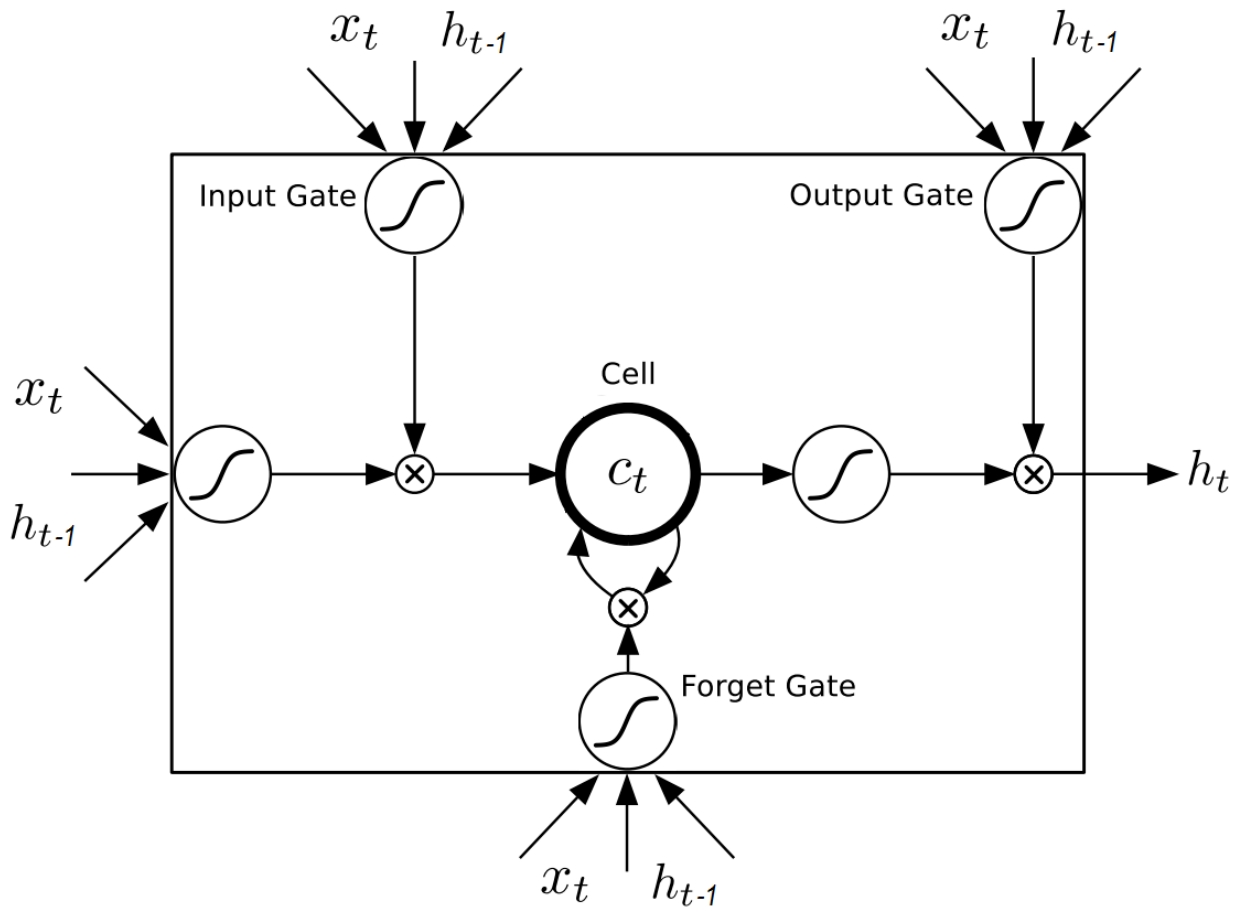


Recurrent Instance Segmentation

- Elements
  - Fully Convolutional Network
  - ConvLSTM
  - Spatial Inhibition
  - Loss function

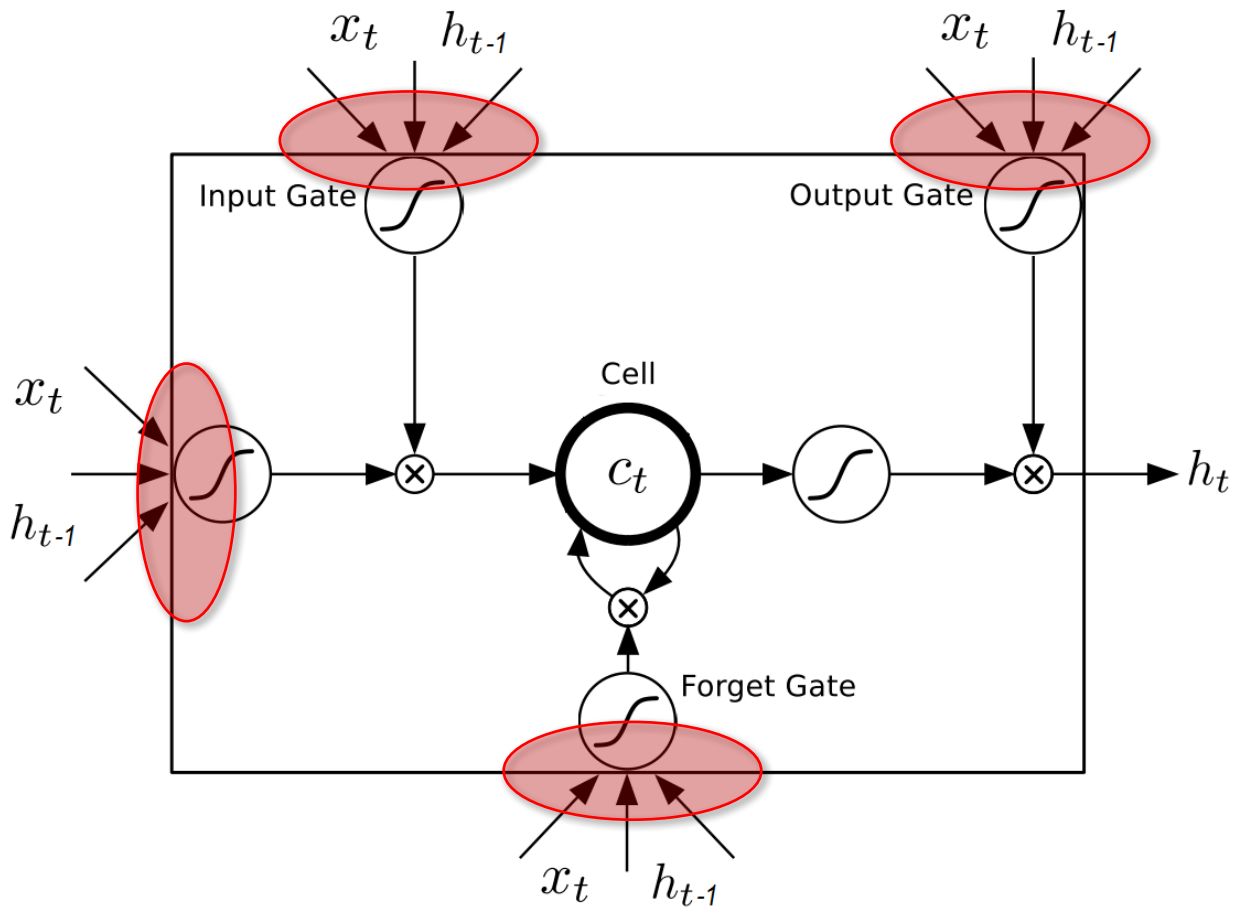


ConvLSTM



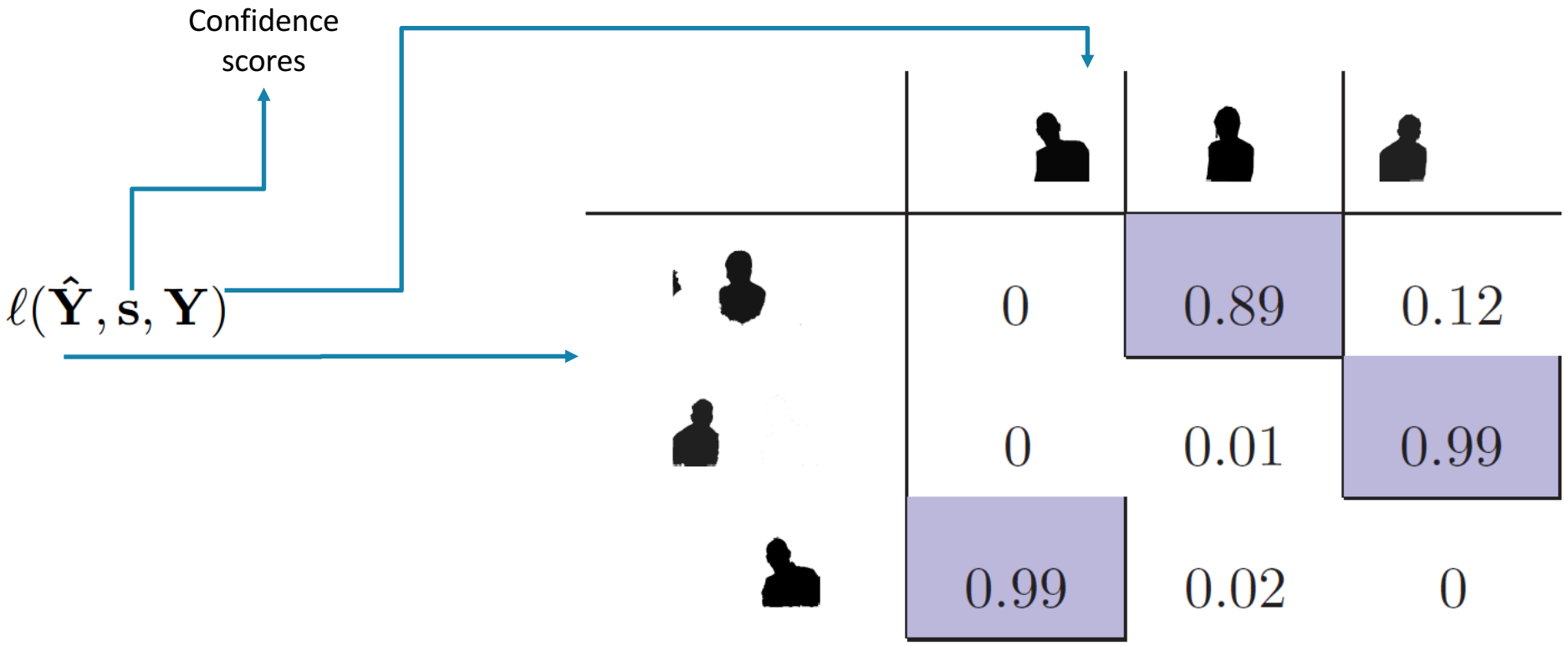
Hochreiter and Schmidhuber 1997

ConvLSTM



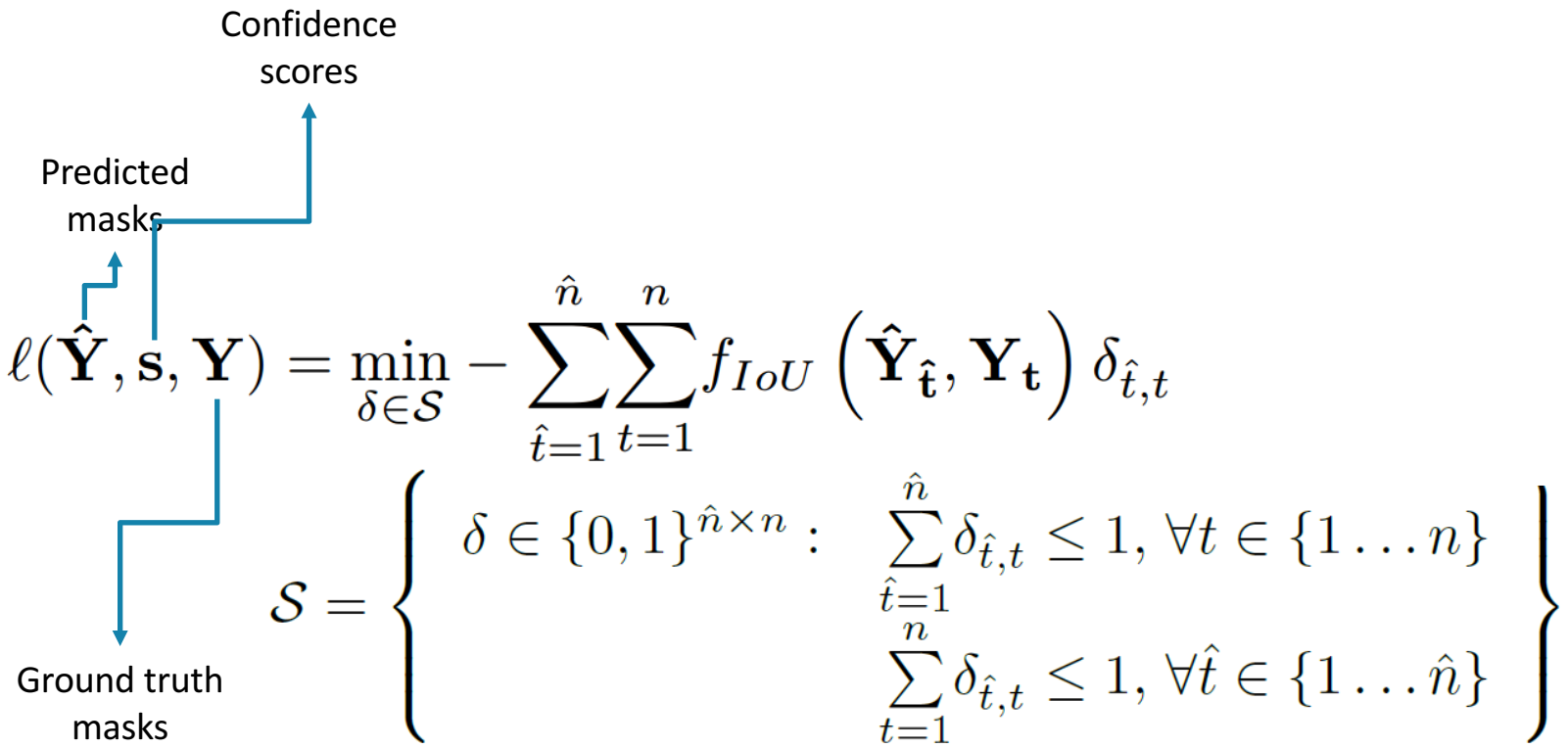
Xingjian et al. 2015

Loss function



Steward & Andriluka, 2015; Silberman et al. 2014

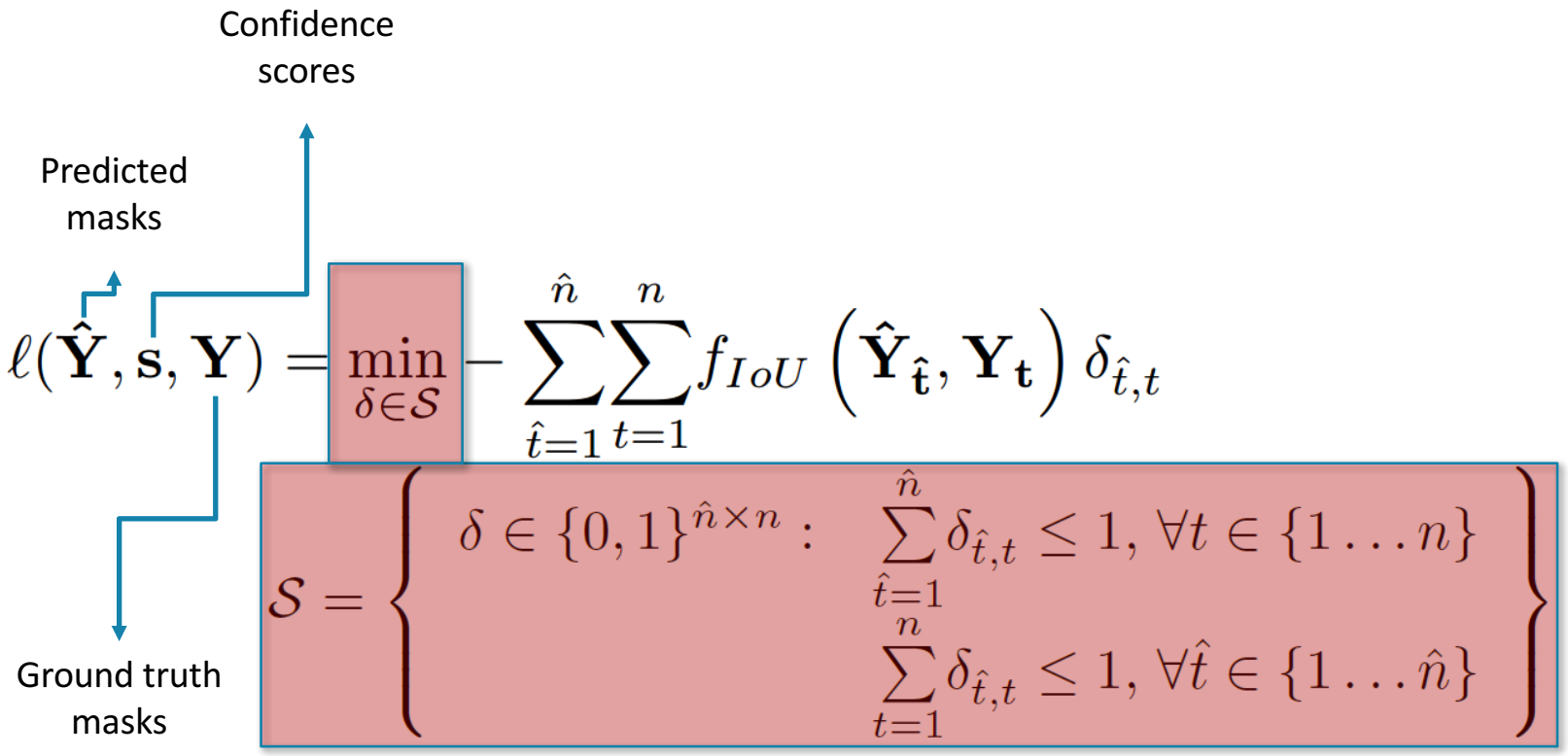
Loss function



$$\ell(\hat{\mathbf{Y}}, \mathbf{s}, \mathbf{Y}) = \min_{\delta \in \mathcal{S}} - \sum_{\hat{t}=1}^{\hat{n}} \sum_{t=1}^{n} f_{IoU}\left(\hat{\mathbf{Y}}_{\hat{\mathbf{t}}}, \mathbf{Y}_{\mathbf{t}}\right) \delta_{\hat{t},t}$$

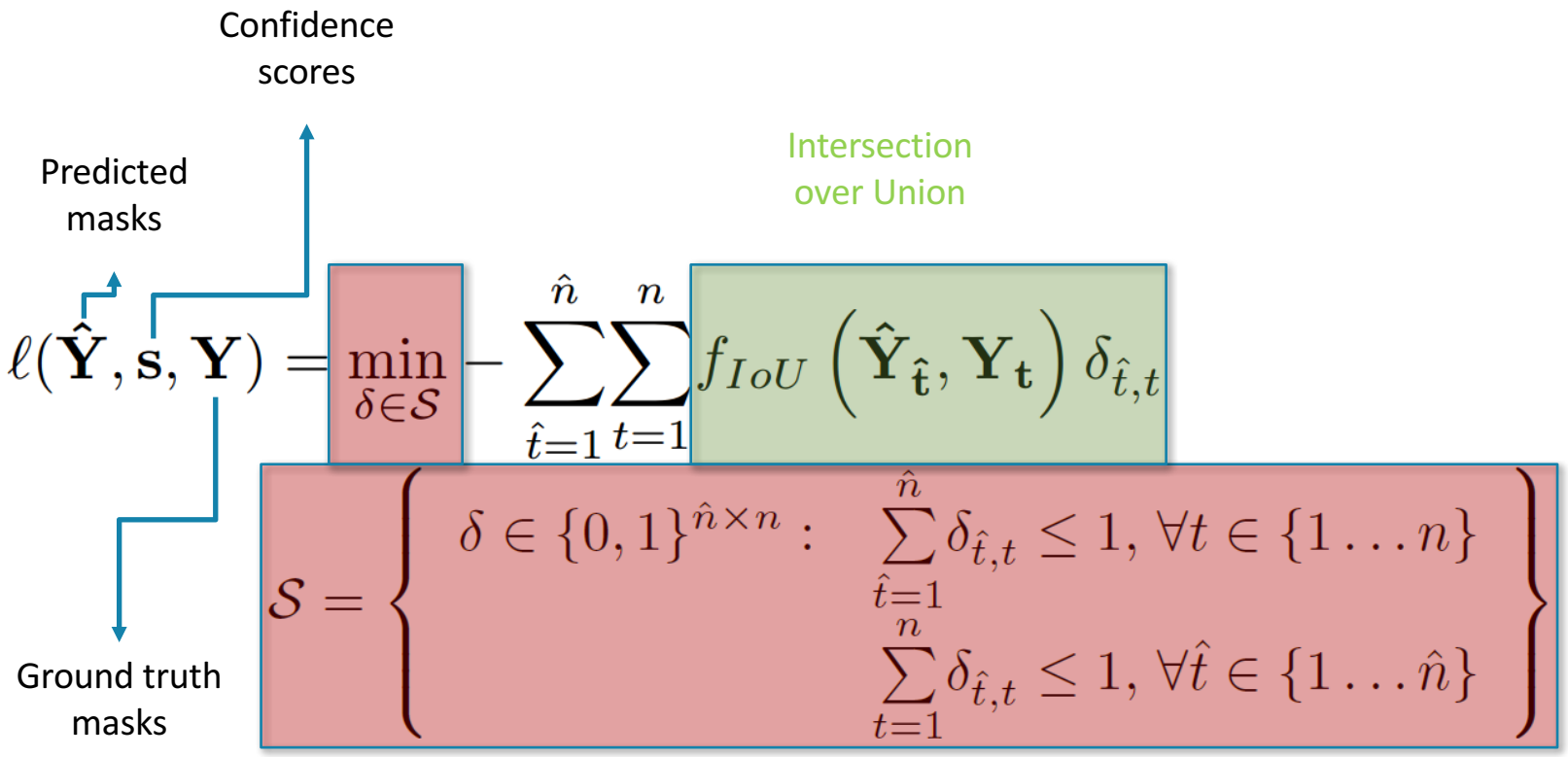
$$\mathcal{S} = \left\{ \begin{array}{ll} \delta \in \{0,1\}^{\hat{n} \times n} : & \sum_{\hat{t}=1}^{\hat{n}} \delta_{\hat{t},t} \leq 1, \forall t \in \{1 \dots n\} \\ & \sum_{t=1}^{n} \delta_{\hat{t},t} \leq 1, \forall \hat{t} \in \{1 \dots \hat{n}\} \end{array} \right\}$$

Loss function



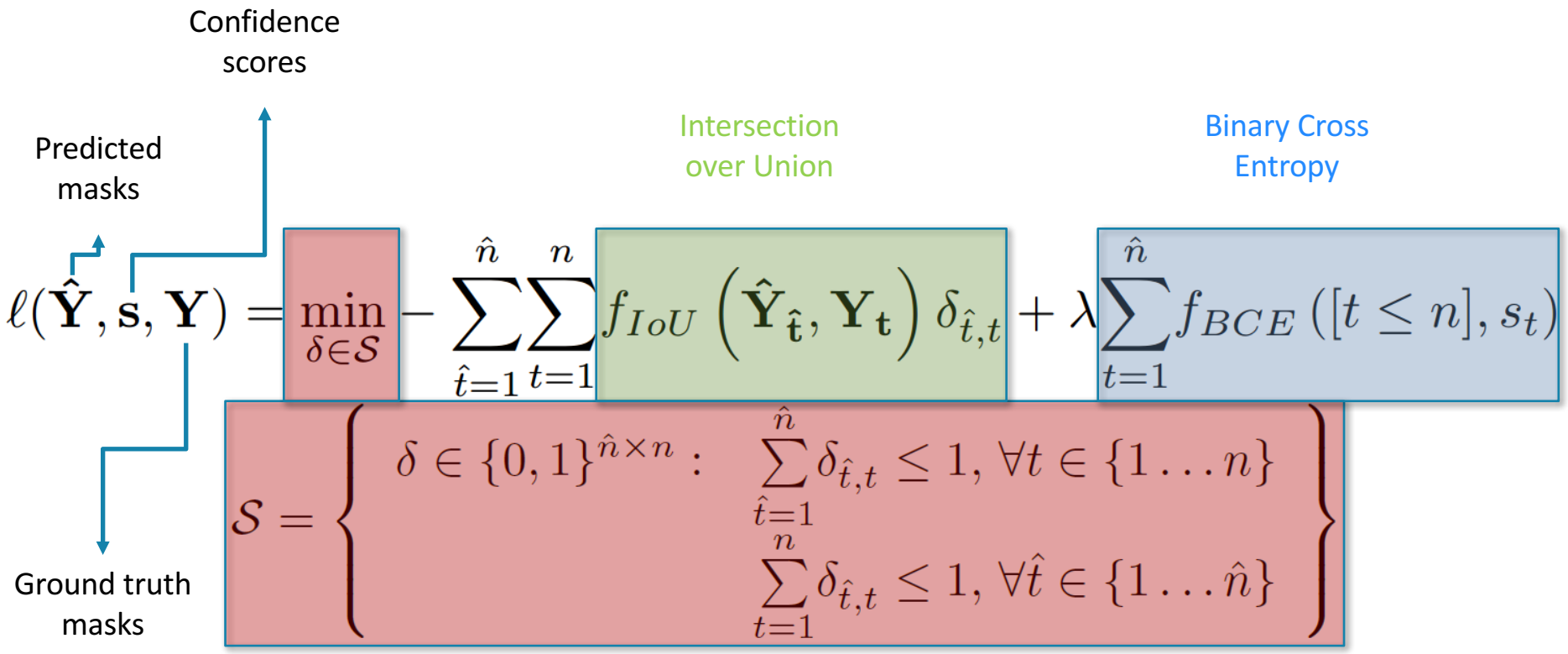
Optimal matching between prediction and ground truth

Loss function



Optimal matching between prediction and ground truth

Loss function

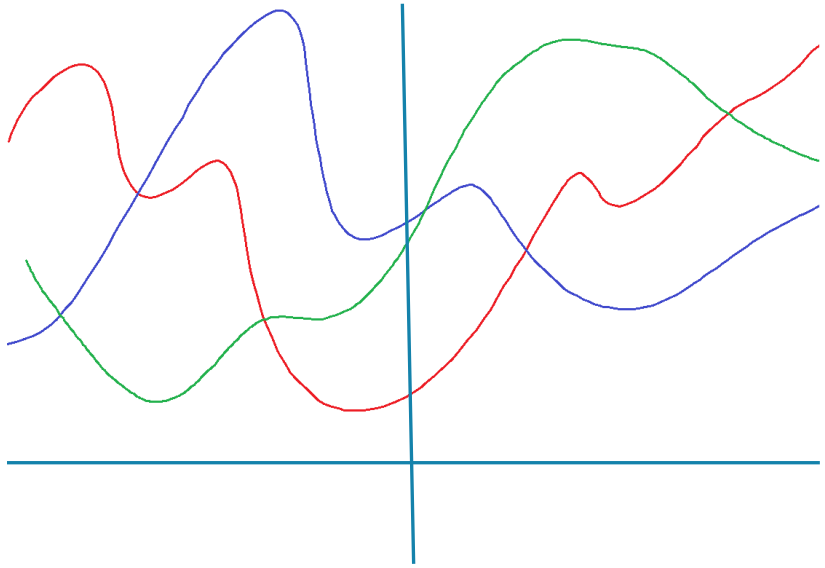


Optimal matching between prediction and ground truth

Loss function

$$\ell(\hat{\mathbf{Y}}, \mathbf{s}, \mathbf{Y}) = \min_{\delta \in \mathcal{S}} - \sum_{\hat{t}=1}^{\hat{n}} \sum_{t=1}^{n} f_{IoU}\left(\hat{\mathbf{Y}}_{\hat{\mathbf{t}}}, \mathbf{Y}_{\mathbf{t}}\right) \delta_{\hat{t},t} + \lambda \sum_{t=1}^{\hat{n}} f_{BCE}\left([t \leq n], s_t\right)$$

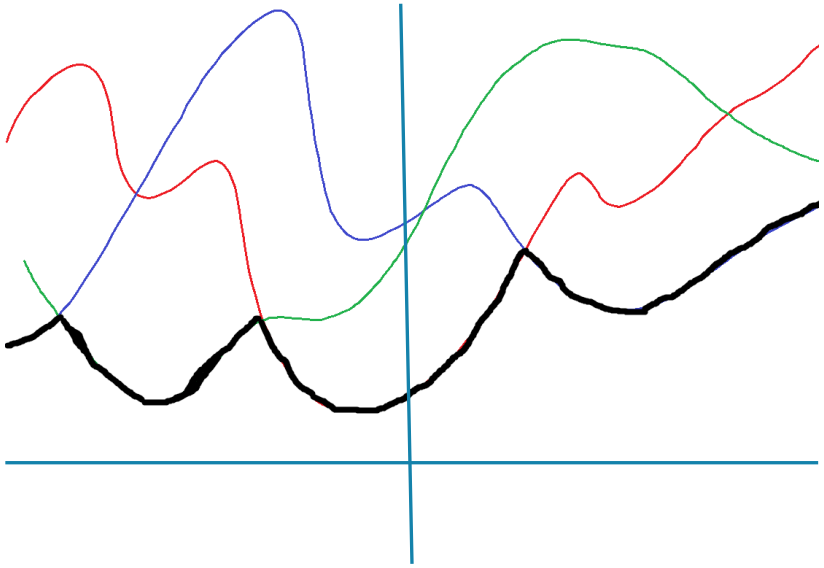
$$\mathcal{S} = \left\{ \delta \in \{0,1\}^{\hat{n} \times n} : \begin{array}{l} \sum_{\hat{t}=1}^{\hat{n}} \delta_{\hat{t},t} \leq 1, \forall t \in \{1 \dots n\} \\ \sum_{t=1}^{n} \delta_{\hat{t},t} \leq 1, \forall \hat{t} \in \{1 \dots \hat{n}\} \end{array} \right\}$$



Loss function

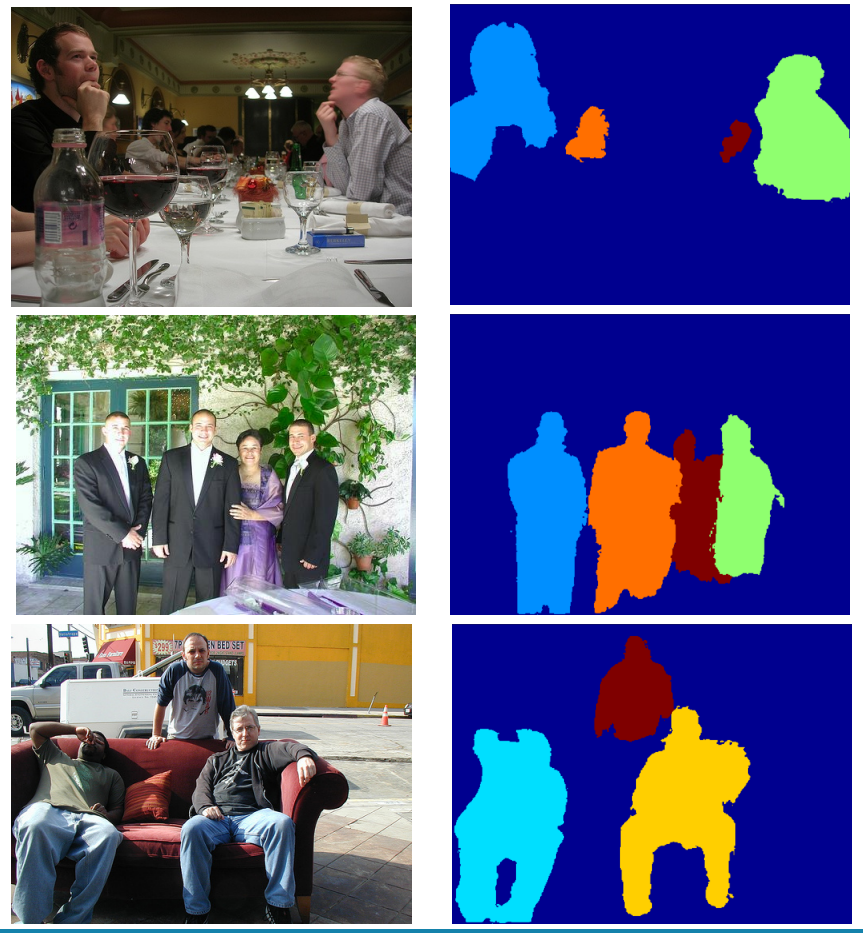
$$\ell(\hat{\mathbf{Y}}, \mathbf{s}, \mathbf{Y}) = \min_{\delta \in \mathcal{S}} - \sum_{\hat{t}=1}^{\hat{n}} \sum_{t=1}^{n} f_{IoU}\left(\hat{\mathbf{Y}}_{\hat{\mathbf{t}}}, \mathbf{Y}_{\mathbf{t}}\right) \delta_{\hat{t},t} + \lambda \sum_{t=1}^{\hat{n}} f_{BCE}\left([t \leq n], s_t\right)$$

$$\mathcal{S} = \left\{ \delta \in \{0,1\}^{\hat{n} \times n} : \begin{array}{l} \sum_{\hat{t}=1}^{\hat{n}} \delta_{\hat{t},t} \leq 1, \forall t \in \{1 \dots n\} \\ \sum_{t=1}^{n} \delta_{\hat{t},t} \leq 1, \forall \hat{t} \in \{1 \dots \hat{n}\} \end{array} \right\}$$



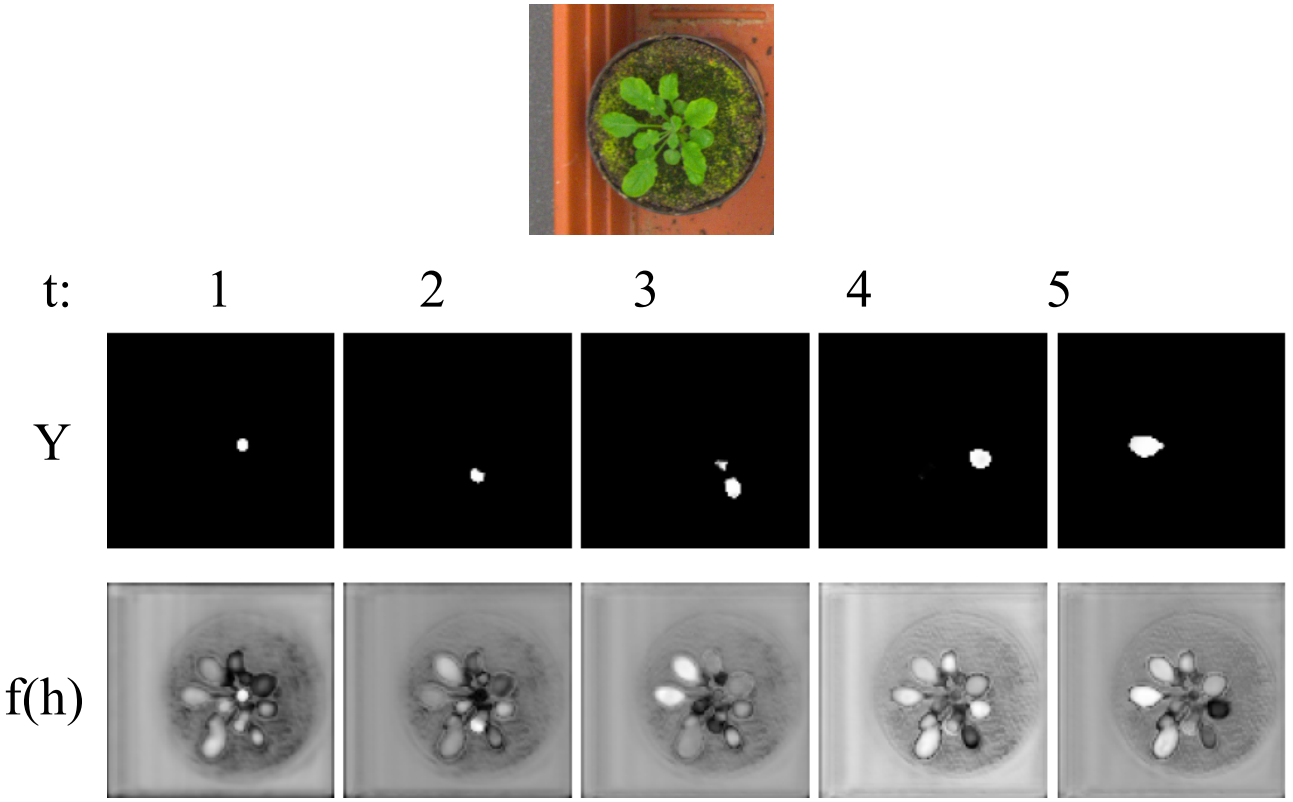
Experiments: Multiple person segmentation

	AP^r (0.5)	AP^r Ave
Chen et al. 2015	48.3	
Hariharan et al. 2014	47.9	
Liang et al. 2015	48.8	42.9
RIS	46.7	41.9
RIS+CRF	**50.1**	**43.7**



Experiments: Leaf counting

	DiC	|DiCl	SBD
IPK	-1.9 (2.5)	2.6 (1.8)	**74.4 (4.3)**
Nottin	3.6 (2.4)	3.8 (2.0)	68.3 (6.3)
MSU	-2.3 (1.6)	2.3 (1.5)	66.7 (7.6)
Wagen	0.4 (3.0)	2.2 (2.0)	71.1 (6.2)
PRIAn	0.8 (1.5)	1.3 (2.0)	
RIS+CRF	**0.2 (1.4)**	**1.1 (0.9)**	66.6 (8.7)



Main points

- Recurrent neural net for instance segmentation
  - It keeps track of visited pixels
  - It can handle occlusion
- Moving beyond:
  - Allowing to classify the segmented instances
  - Recurrent attention [Ren and Zemel, 2016]

Collaborators

- Dr. Sadeep Jayasumana
- Zhizong Su
- Dalong Du
- Dr. Chang Huang
- Dr. Vibhav Vineet
- Prof. Philip H.S Torr

Acknowledgements

Engineering and Physical Sciences Research Council

EPSRC EP/I001107/2

European Research Council

ERC- 2012-AdG 321162-HELIOS

Thank you

Try your photo on our award-winning demo

http://crfasrnn.torr.vision

This demo won best demo award in ICCV 2015

Twitter #SegmentYourPhoto

Original image (hover to highlight segmented parts)

Objects appearing in the image:

Bird Dog Person

Semantic segmentation

Questions?

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H.S. Torr. Conditional Random Fields as Recurrent Neural Networks. ICCV, 2015

Anurag Arnab, Sadeep Jayasumana, Shuai Zheng, Philip H.S. Torr. Higher Order Potentials in End-to-End Trainable Conditional Random Fields. ECCV, 2016

Anurag Arnab, Philip H.S. Torr. Bottom-up Instance Segmentation using Deep Higher-Order CRFs. BMVC, 2016.

Bernardino Romera-Paredes, Philip H.S. Torr. Recurrent Instance Segmentation. ECCV, 2016.