Recurrent neural network and conditional random field

Outline

- Recurrent neural network (RNN)
- Graphical model
- Connecting graphical model and RNN

Outline

- Recurrent neural network (RNN)
- Graphical model
- Connecting graphical model and RNN

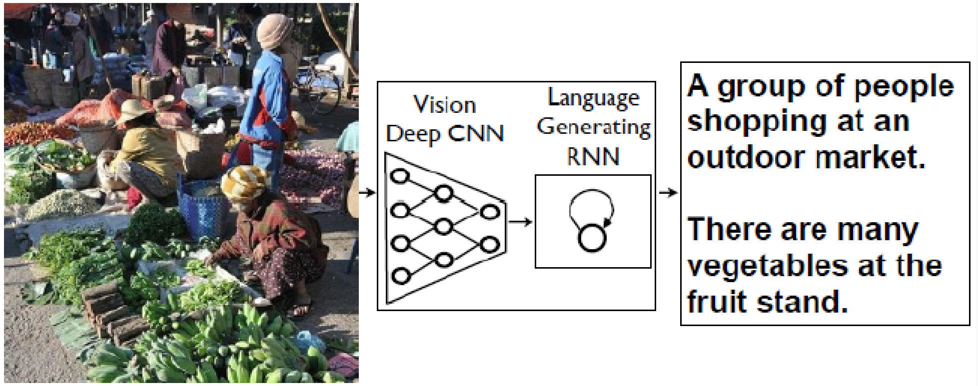
Modelling sequential data

- Sample data sequences from a certain distribution

$$P(\mathbf{x}_1, \ldots, \mathbf{x}_T)$$

- Generate natural sentences to describe an image

$$P(\mathbf{x}_1, \ldots, \mathbf{x}_T \,|\mathbf{I})$$



- Activity recognition from a video sequence

$$P(y|\mathbf{x}_1, \ldots, \mathbf{x}_T)$$

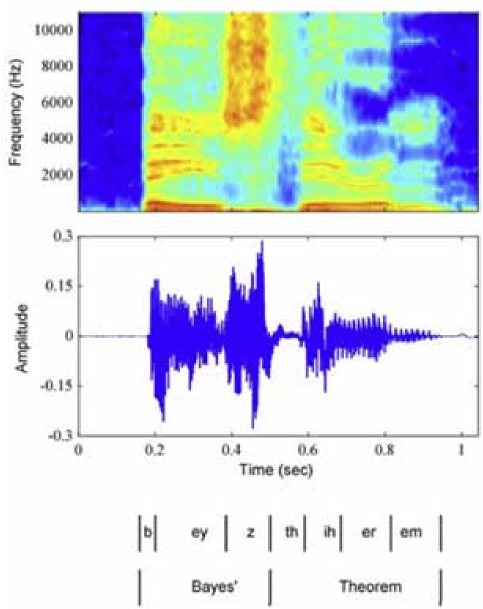
Modelling sequential data

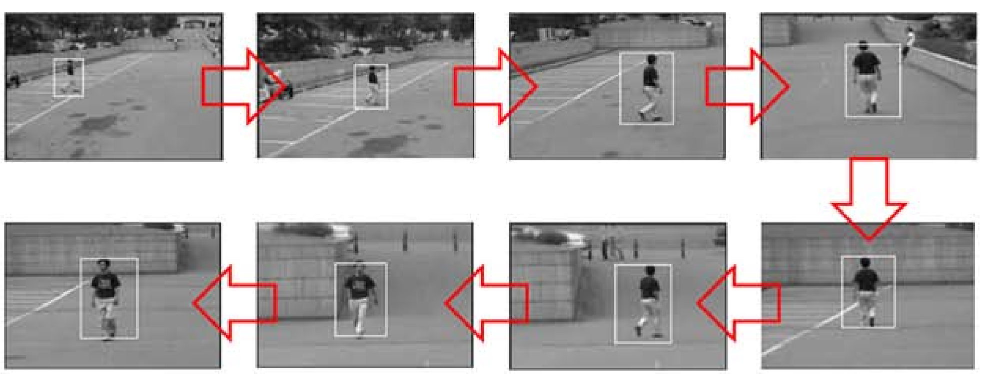
- Speech recognition

$$P(\mathbf{y}_1, \ldots, \mathbf{y}_T | \mathbf{x}_1, \ldots, \mathbf{x}_T)$$

- Object tracking

$$P(\mathbf{y}_1, \ldots, \mathbf{y}_T | \mathbf{x}_1, \ldots, \mathbf{x}_T)$$





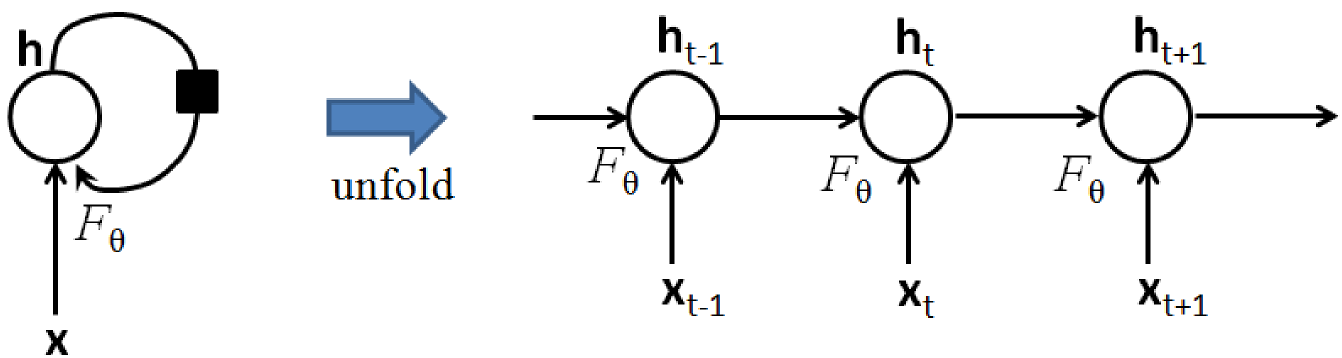
Recurrent neural networks (RNN)

- Model a dynamic system driven by an external signal $\mathbf{x}_t$

$$\boldsymbol{h}_t = F(\boldsymbol{h}_{t\text{-}1}; \mathbf{x}_t)$$

- $\boldsymbol{h}_t$ contains information about the whole past sequence. The equation above implicitly defines a function which maps the whole past sequence $(\mathbf{x}_\mathrm{t}, \ldots, \mathbf{x}_1)$ to the current state

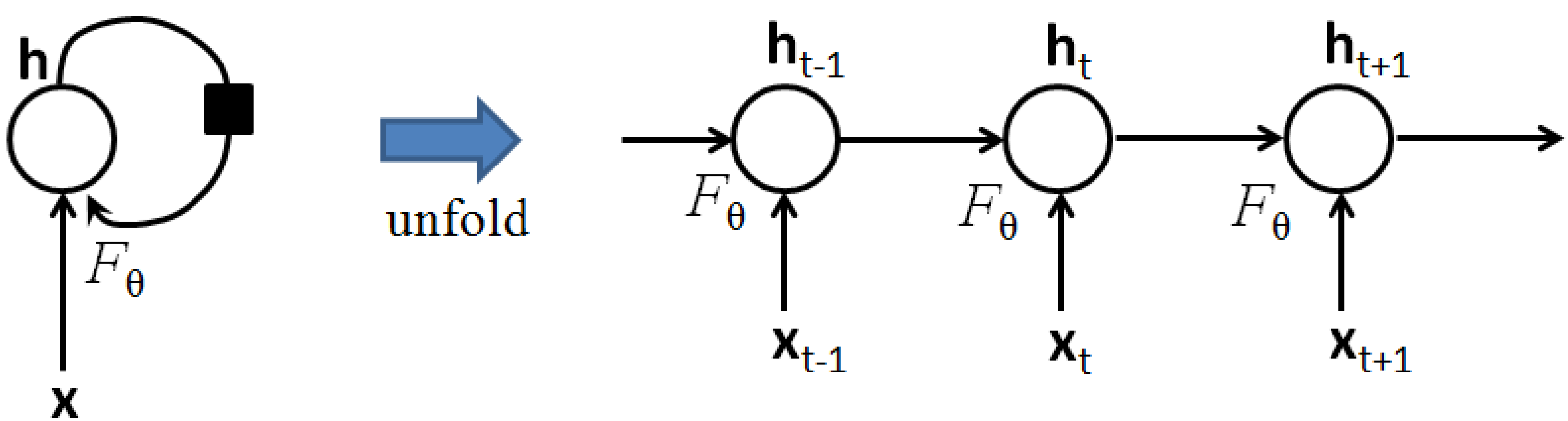
$$\boldsymbol{h}_t = G_t(\mathbf{x}_\mathrm{t}, \ldots, \mathbf{x}_1).$$



Left: physical implementation of RNN, seen as a circuit. The black square indicates a delay of 1 time step. Right: the same seen as an unfolded flow graph, where each node is now associated with one particular time

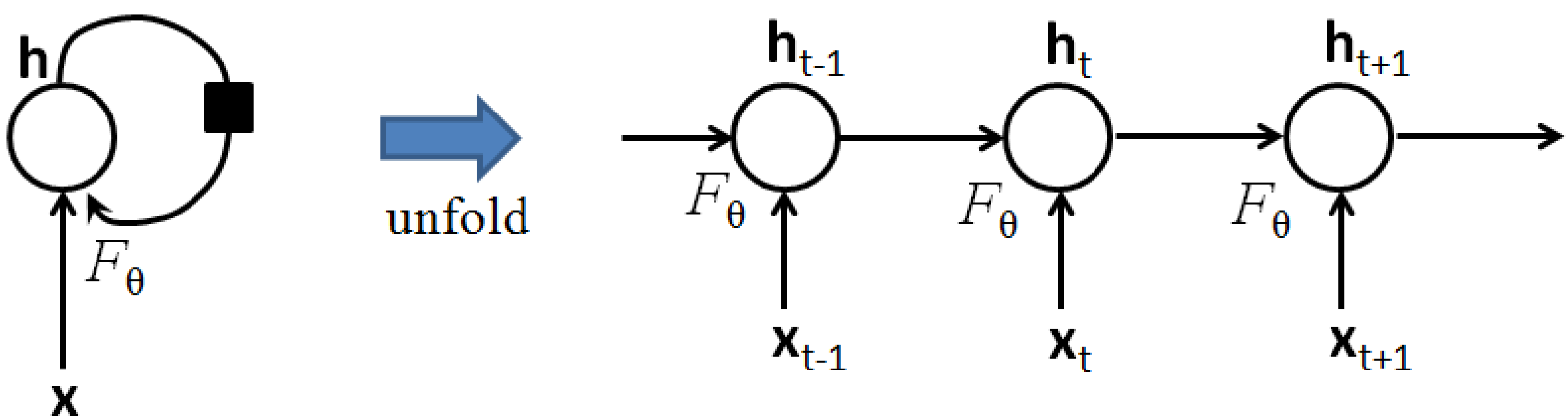
Recurrent neural networks (RNN)

- Sharing parameters: the same weights are used for different instances of the artificial neurons at different time steps
- Share a similar idea with CNN: replacing a fully connected network with local connections with parameter sharing
- It allows to apply the network to input sequences of different lengths and predict sequences of different lengths



Recurrent neural networks (RNN)

- **Sharing parameters for any sequence length allows better generalization properties.** If we have to define a different function G_t for each possible sequence length, each with its own parameters, we would not get any generalization to sequences of a size not seen in the training set. One would need to see a lot more training examples, because a separate model would have to be trained for each sequence length.



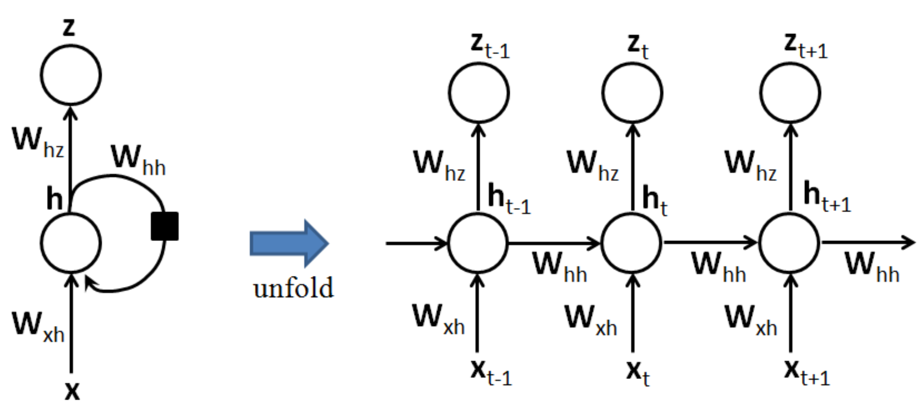
A vanilla RNN to predict sequences from input

- $P(\mathbf{z}_1, \ldots, \mathbf{z}_T | \mathbf{x}_1, \ldots, \mathbf{x}_T)$
- Forward propagation equations, assuming that hyperbolic tangent non-linearities are used in the hidden units and softmax is used in output for classification problems

$$\mathbf{h}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t\text{-}1} + \mathbf{b}_h)$$

$$\mathbf{z}_t = \text{softmax}(\mathbf{W}_{hz}\mathbf{h}_t + \mathbf{b}_z)$$

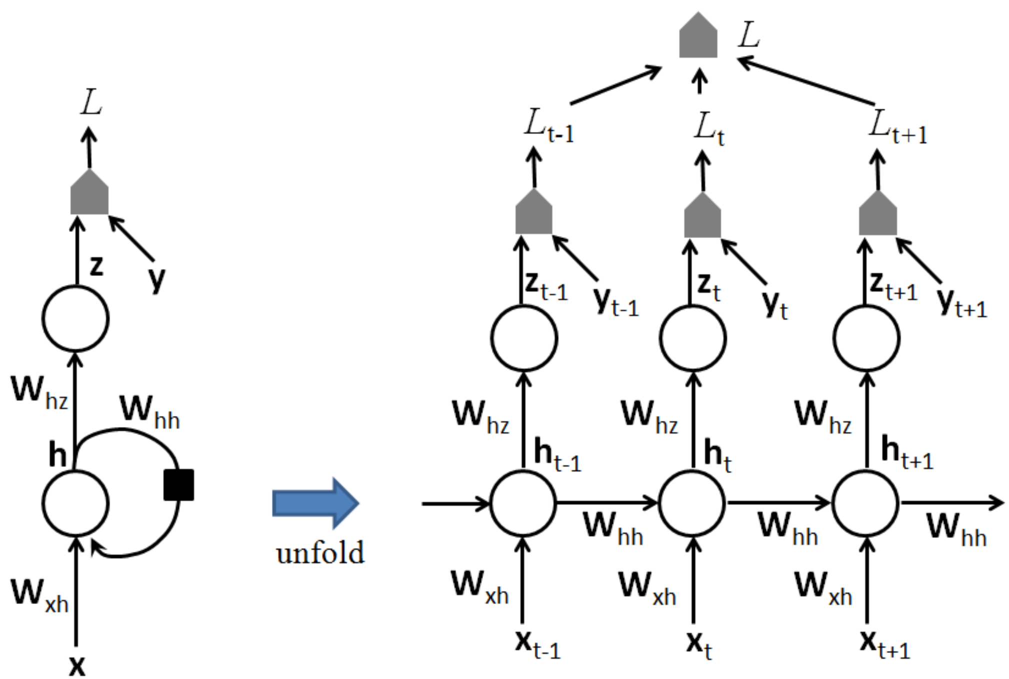
$$p(\mathrm{z}_t = c) = z_{t,c}$$



Loss function

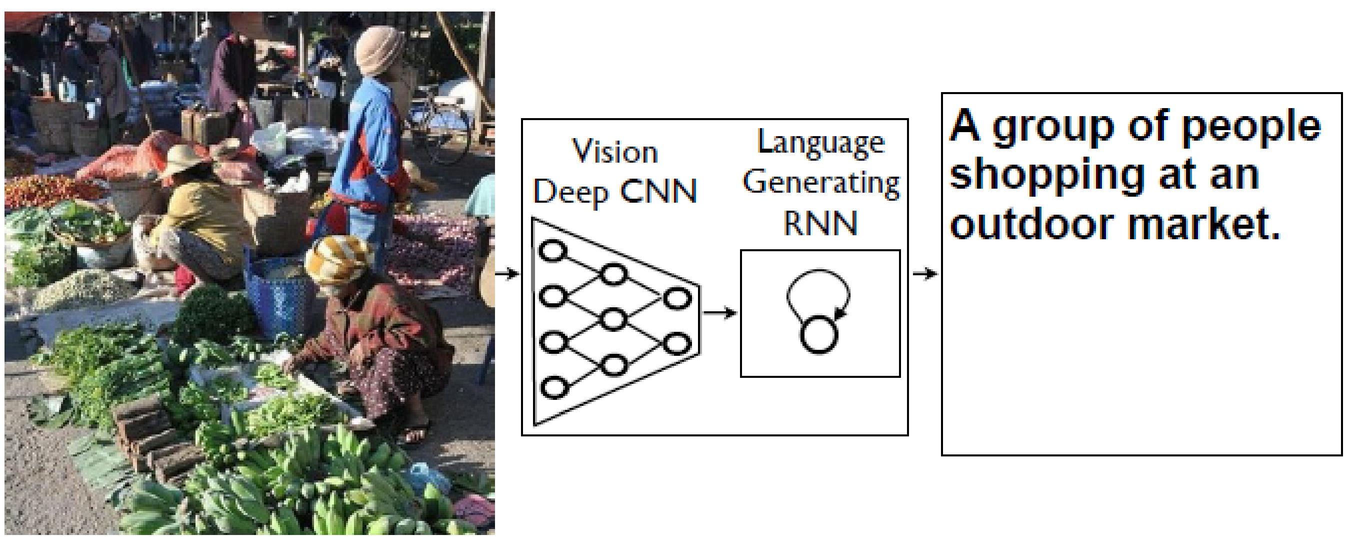
- The total loss for a given input/target sequence pair (**x**, **y**), measured in cross entropy

$$L(\mathbf{x}, \mathbf{y}) = \sum_t L_t = \sum_t -\log z_{t,y_t}$$



Application of RNN - image captioning

- Generate image caption Vinyals et al. arXiv 2014
- Use a CNN as an image encoder and transform it to a fixed-length vector
- It is used as the initial hidden state of a “decoder” RNN that generates the target sequence



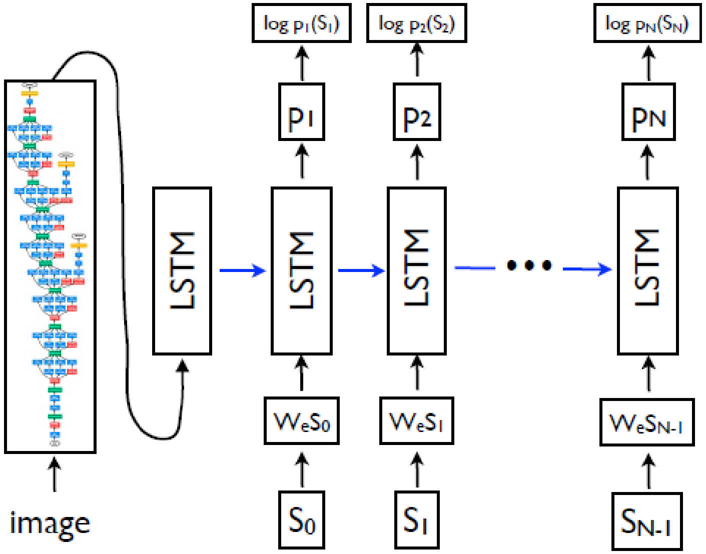
Application of RNN - image captioning

- The learning process is to maximize the probability of the correct description given the image

$$\theta^* = \arg\max \sum_{(\mathbf{I},\mathbf{S})} \log P(\mathbf{S}|\mathbf{I};\theta)$$

$$\log P(\mathbf{S}|\mathbf{I}) = \sum_{t=0}^{N} \log P(\mathbf{S}_t|I,\mathbf{S}_0,\ldots,\mathbf{S}_{t-1})$$

- $\mathbf{I}$ is an image and $\mathbf{S}$ is its correct description



Application of RNN - image captioning

- Denote by $\mathbf{S}_0$ a special start word and by $\mathbf{S}_N$ a special stop word
- Both the image and the words are mapped to the same space, the image by using CNN, the words by using word embedding $\mathbf{W}_e$
- The image $\mathbf{I}$ is only input once at $t = -1$ to inform the LSTM (an RNN implementation) about the image contents
- Sampling: sample the first word according to P_1, then provide the corresponding embedding as input and sample P_2, continuing like this until it samples the special end-of-sentence token

$$\mathbf{x}_{-1} = \mathrm{CNN}(\mathbf{I})$$

$$\mathbf{x}_t = \mathbf{W}_e \mathbf{S}_t, t \in \{0, \ldots, N-1\}$$

$$P_{t+1} = \mathrm{LSTM}(\mathbf{x}_t), t \in \{0, \ldots, N-1\}$$

$$L(\mathbf{I}, \mathbf{S}) = -\sum_{t=1}^{N} \log P_t(\mathbf{S}_t)$$

Last time

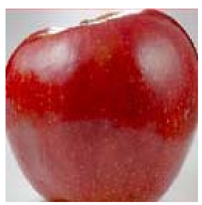
- Interesting presentation:
- https://www.youtube.com/watch?v=40riCqvRoMs
- Interesting demo:
- https://www.youtube.com/watch?v=OOT3UIXZztE
- https://www.youtube.com/watch?v=B3omChYHaoo
- https://www.youtube.com/watch?v=Mc_31ZPRm9g

Outline

- Recurrent neural network (RNN)
- Graphical model
- Connecting graphical model and RNN

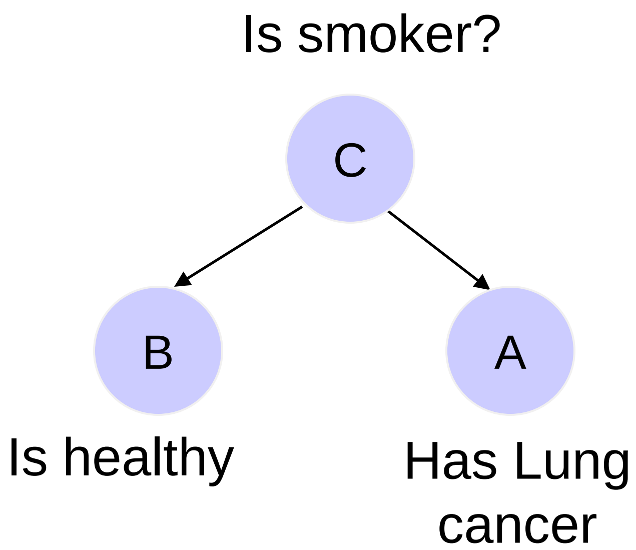
Graphical model for Statistics

- Use graph to represent the joint distribution of random variables P(Image, label)
- Example:

( , *Apple*) ( , *Pear*)

Graphical model for Statistics

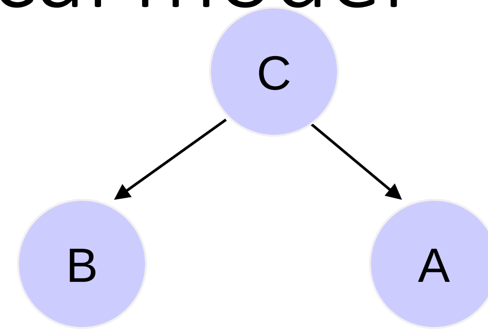
- Conditional independence between random variables
- Given C, A and B are independent:
  - $P(A, B|C) = P(A|C)P(B|C)$
- $P(A,B,C) = \mathbf{P(A, B|C)}\ P(C)$
  - $= \mathbf{P(A|C)P(B|C)}P(C)$
  - Any two nodes are conditionally independent given the values of their parents.



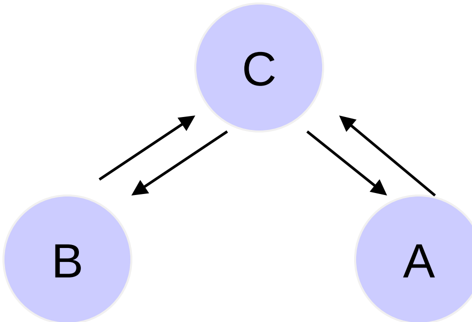
"Has Lung cancer" and "Is healthy" are not independent.
$P(A,B) \neq P(A)P(B)$
However, knowing that the person is a smoker, they are independent.

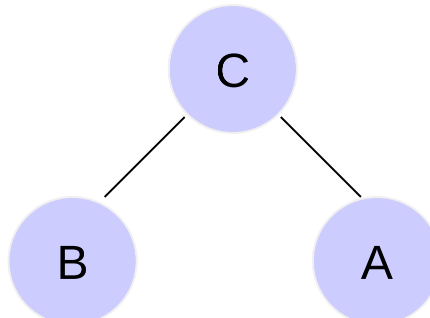
http://www.eecs.qmul.ac.uk/~norman/BBNs/Independence_and_conditional_independence.htm

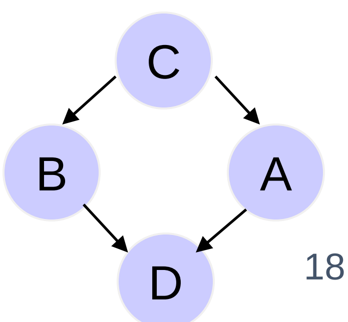
Directed and undirected graphical model



- Directed graphical model
  - P(A,B,C) = P(A|C)P(B|C)P(C)
  - Any two nodes are conditionally independent given the values of their parents.
- Undirected graphical model
  - Clique: fully connected sub-graph Φ
  - P(A,B,C) = Φ(B,C)Φ(A,C)
  -Also called Marcov Random Field (MRF)



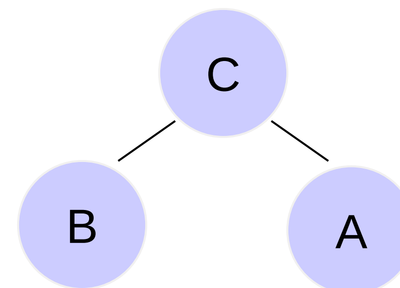




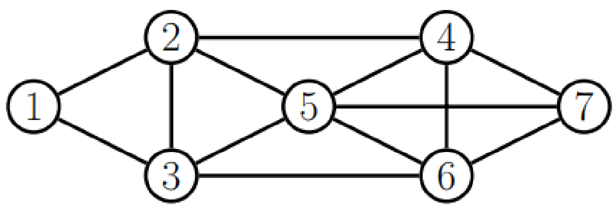
P(A,B,C,D) = P(D|A,B)P(B|C)P(A|C)P(C)

Modeling undirected model

- Two nodes are conditionally independent given the rest if there is no direct edge between them:
- Example 1:
  $A \perp B \mid C$
- Example 2:
  $1 \perp 7 \mid \text{rest}$
  $1 \perp \text{rest} \mid 2, 3$

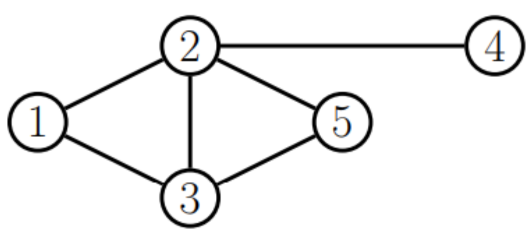


Example 1



Example 2

- Example 3:
  $3 \perp 4 \mid 2$



Example 3

Modeling undirected model

- Probability representation
- A positive distribution p(y) > 0 satisfies the CI properties of an undirected graph G *iff* p can be represented as a product of factors, one per maximal clique, i.e.,

$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \varphi(\mathbf{x}_c|\theta_c)$$

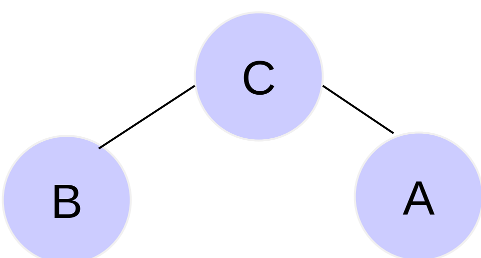
where *C* is the set of all the (maximal) cliques of G, and Z(θ) is the partition function given by

$$Z(\theta) = \sum_{\mathbf{x}} \prod_{c \in C} \varphi(\mathbf{x}_c|\theta_c)$$

Modeling undirected model

- Probability representation
- A positive distribution p(y) > 0 satisfies the CI properties of an undirected graph G *iff* p can be represented as a product of factors, one per maximal clique, i.e.,

$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \varphi(\mathbf{x}_c|\theta_c)$$

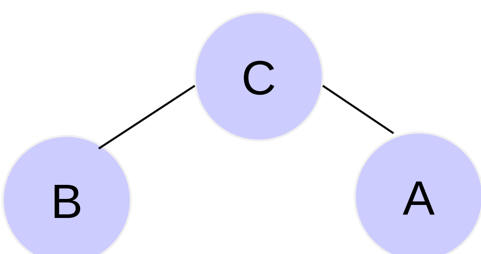


$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \varphi(A,C)\ \varphi(B,C)$$

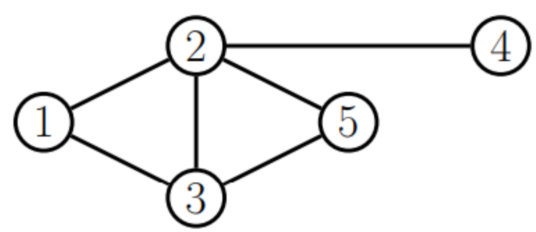
Modeling undirected model

- Probability representation
- A positive distribution p(y) > 0 satisfies the CI properties of an undirected graph G *iff* p can be represented as a product of factors, one per maximal clique, i.e.,

$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \varphi(\mathbf{x}_c | \theta_c)$$



$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \varphi(A, C)\ \varphi(B, C)$$



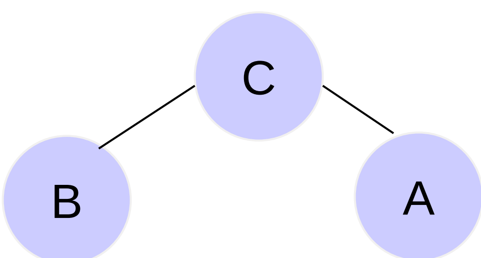
$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \varphi(1,2,3)\ \varphi(2,3,5)\ \varphi(2,4)$$

Modeling undirected model

- Probability representation
- A positive distribution p(y) > 0 satisfies the CI properties of an undirected graph G *iff* p can be represented as a product of factors, one per maximal clique, i.e.,

$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \prod_{c \in C} \varphi(\mathbf{x}_c|\theta_c) = \frac{1}{Z(\theta)} e^{\sum_{c\in C} \log \varphi(\mathbf{x}_c|\theta_c)} = \frac{1}{Z(\theta)} e^{-E(\mathbf{x};\theta)}$$

$$E(\mathbf{x};\theta) = -\sum_{c\in C} \log \varphi(\mathbf{x}_c|\theta_c)$$



$$P(\mathbf{x};\theta) = \frac{1}{Z(\theta)} \varphi(A,C)\, \varphi(B,C)$$

$$P(A,B,C;\theta) = \frac{\exp(w_1 BC + w_2 AC)}{\sum_{A,B,C} \exp(w_1 BC + w_2 AC)}$$

$$E(\mathrm{x};\theta) = -(w_1 BC + w_2 AC)$$

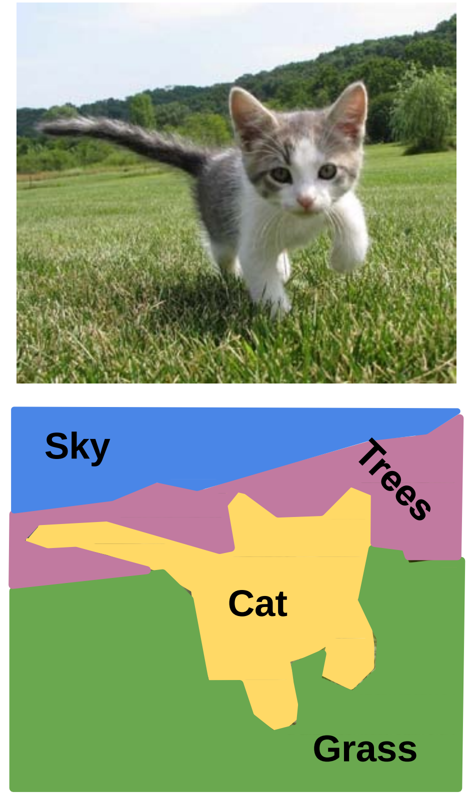
Outline

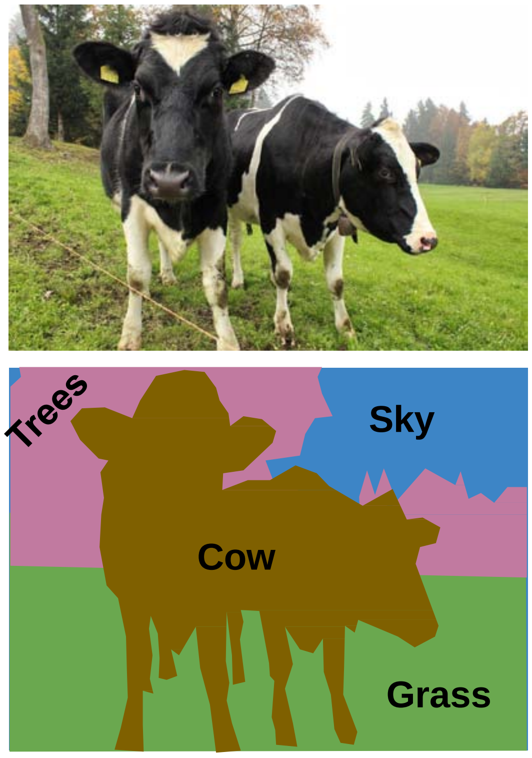
- Recurrent neural network (RNN)
- Graphical model
- Connecting graphical model and RNN

Semantic Segmentation

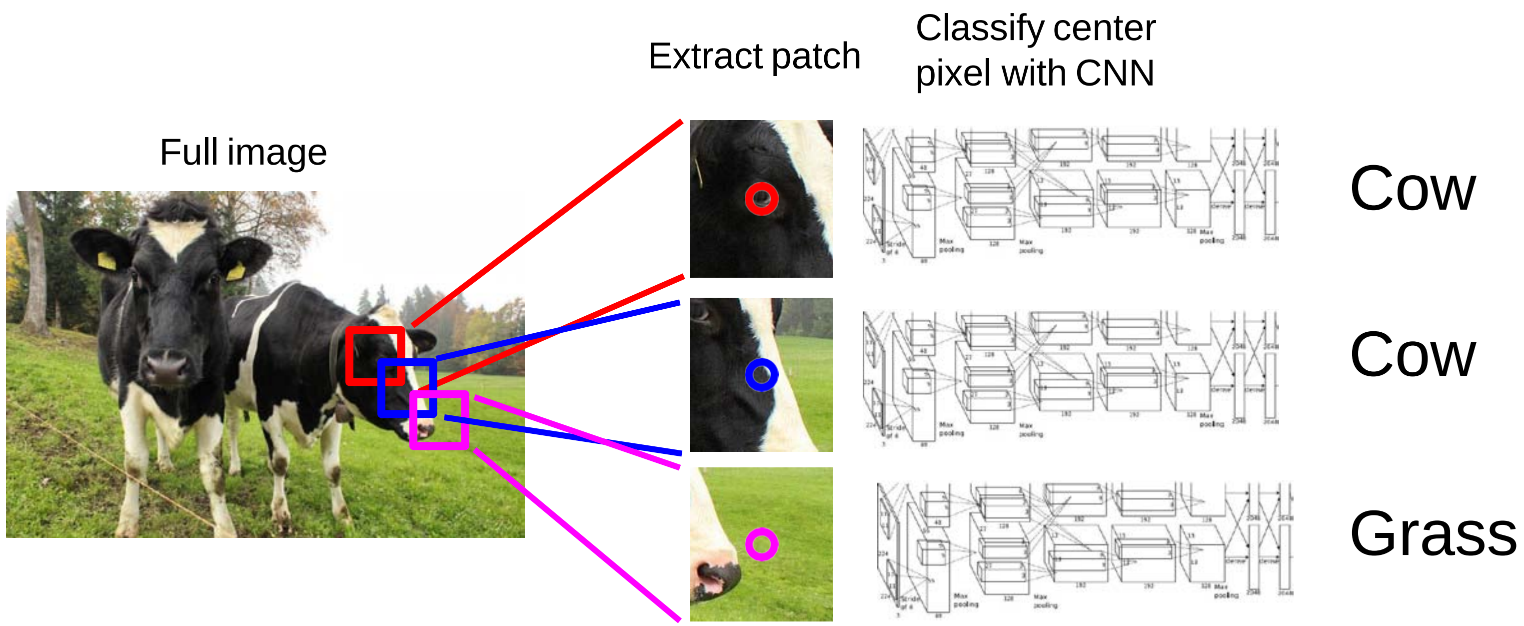
Label each pixel in the image with a category label

Don't differentiate instances, only care about pixels



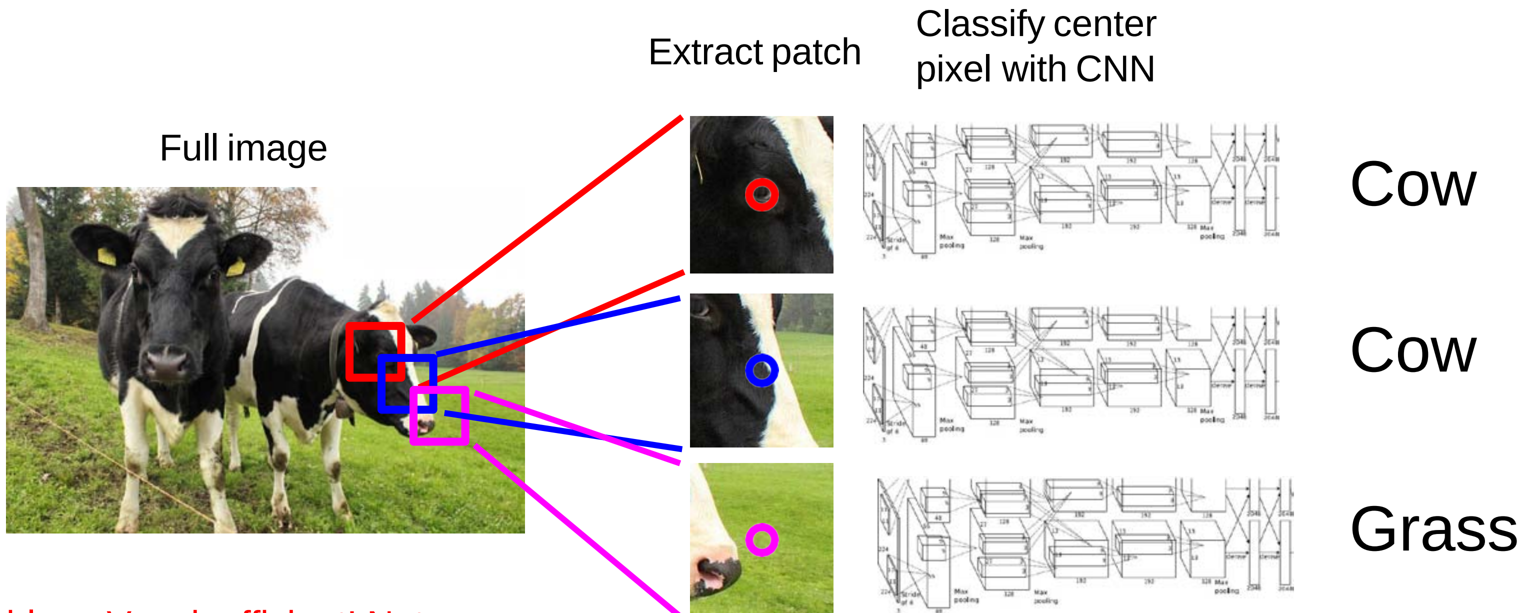


Semantic Segmentation Idea: Sliding Window



Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI 2013
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014

Semantic Segmentation Idea: Sliding Window

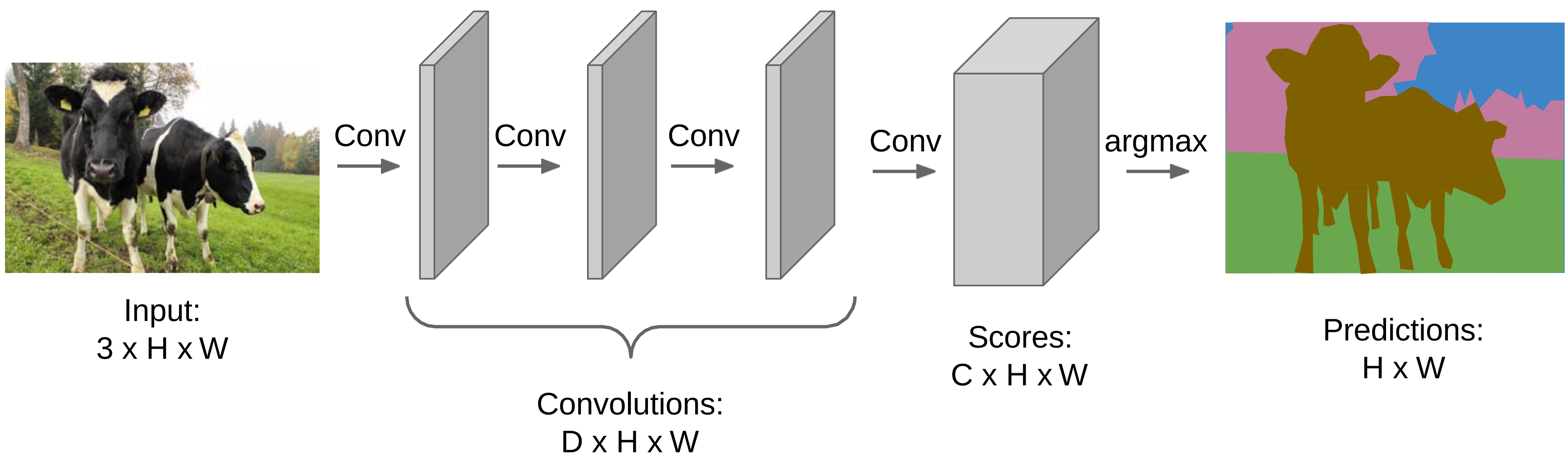


Problem: Very inefficient! Not reusing shared features between overlapping patches

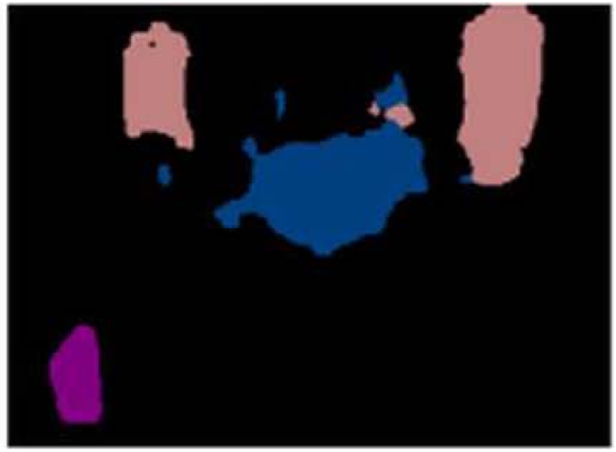
Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI 2013
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014

Semantic Segmentation Idea: Fully Convolutional

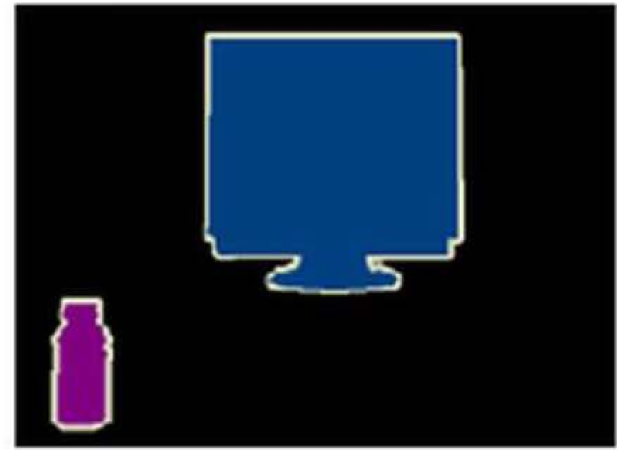
Design a network as a bunch of convolutional layers
to make predictions for pixels all at once!



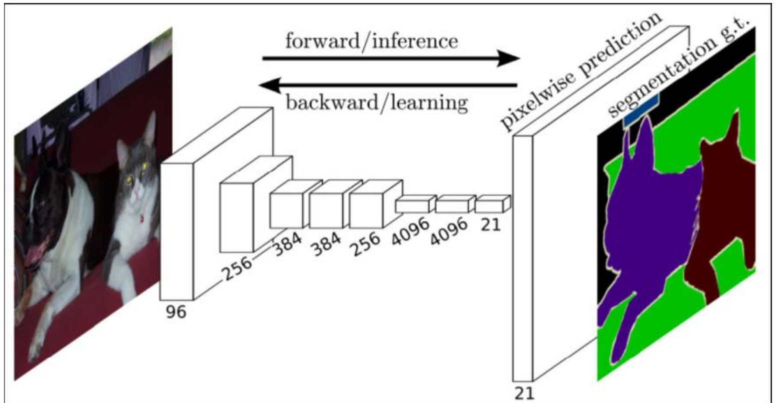
Semantic Segmentation



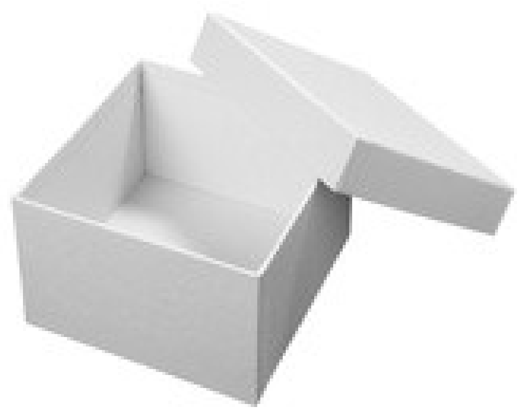
Fully convolutional networks
[Long et al. CVPR 2015]



Ground truth

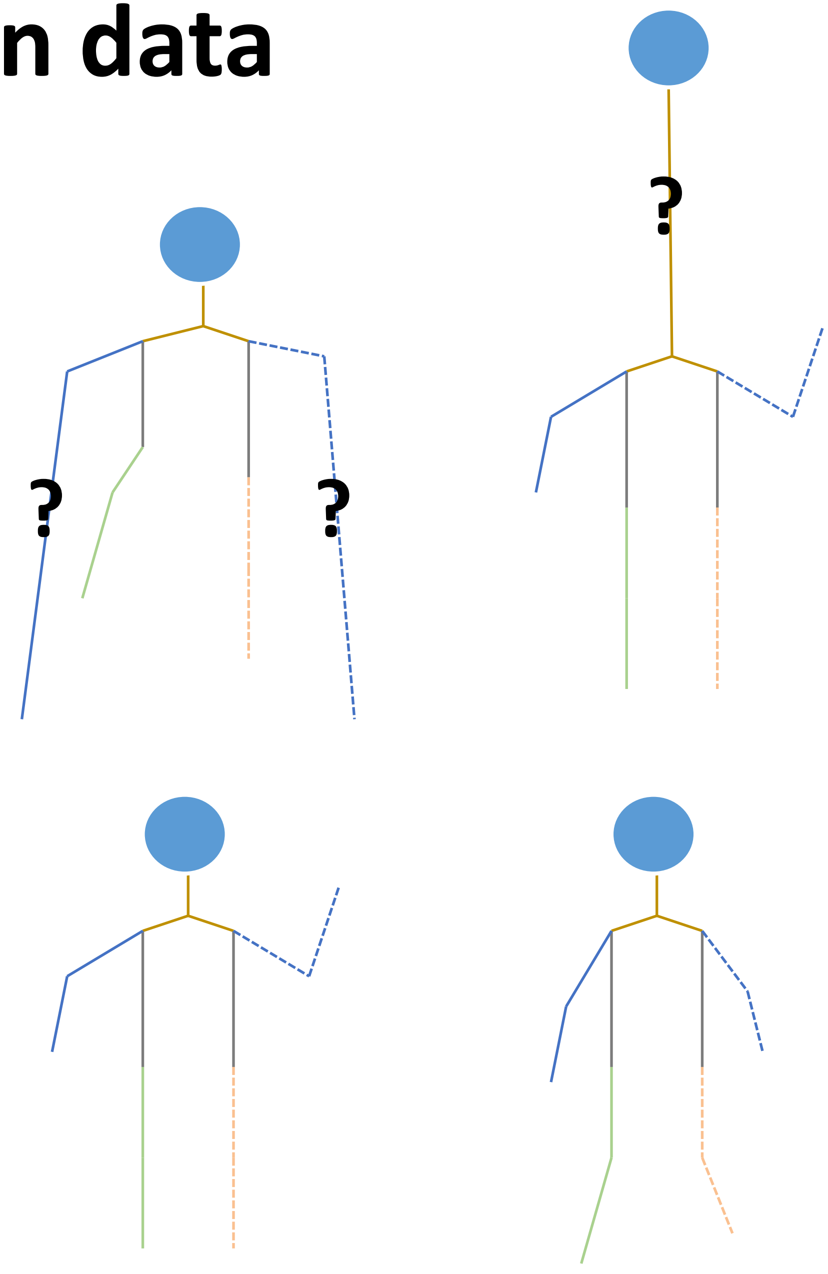


Is deep model a black box?

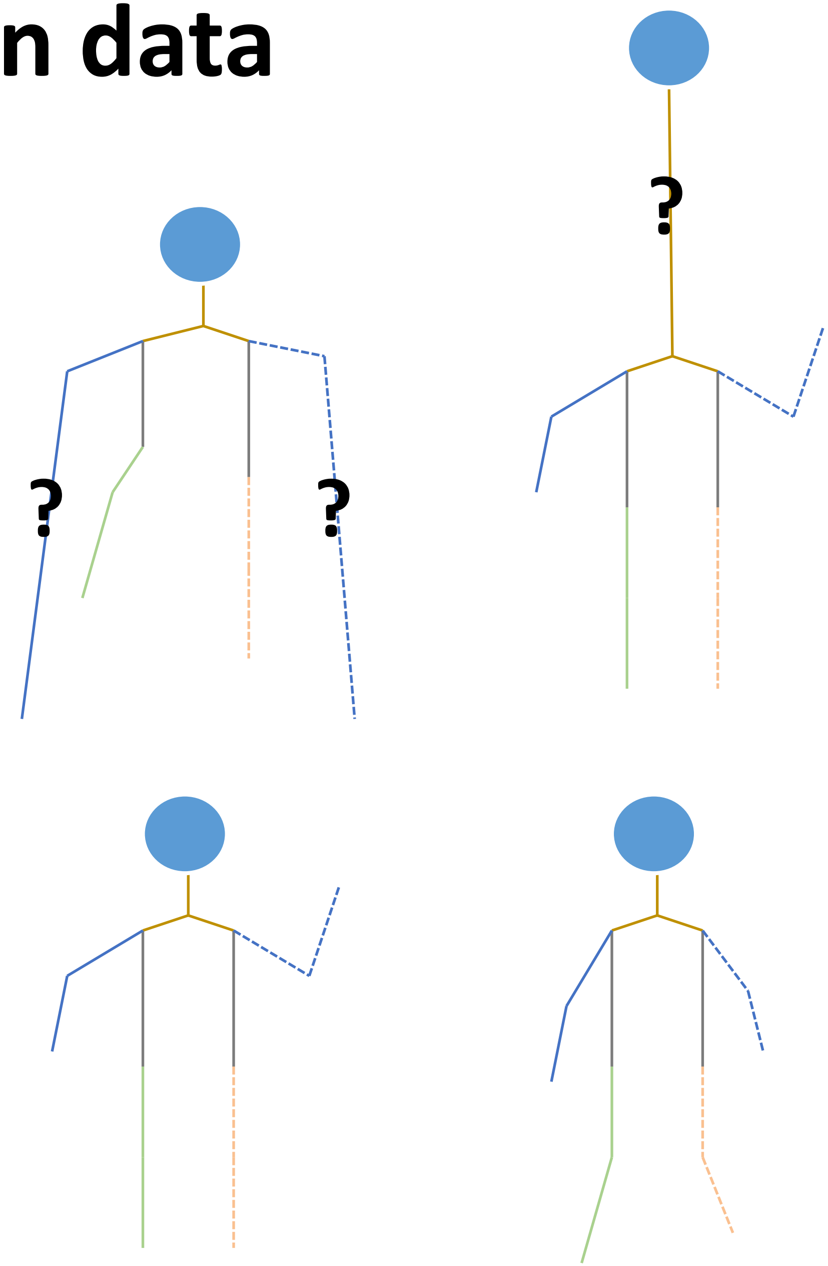


Structure in data

Structure in data

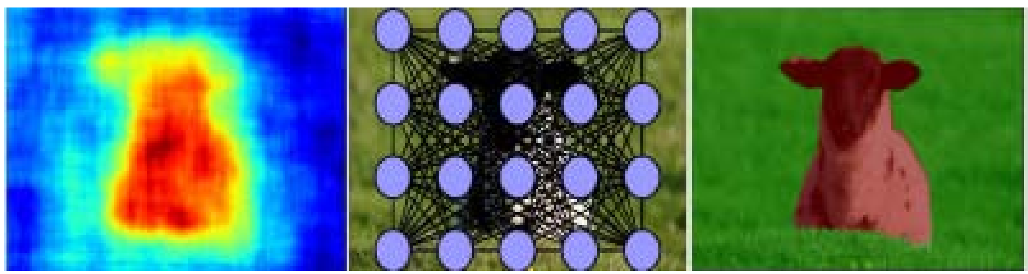


Structure in data

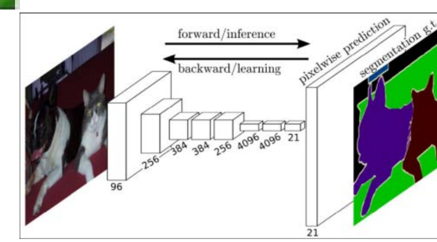


Conditional Random Fields (CRFs)

- A conditional random field is simply a conditional distribution $P(\mathbf{X}|\mathbf{I})$ with an associated graphical structure.



$$P(\mathbf{X}|\mathbf{I}) :$$



$\mathbf{I}$: Image

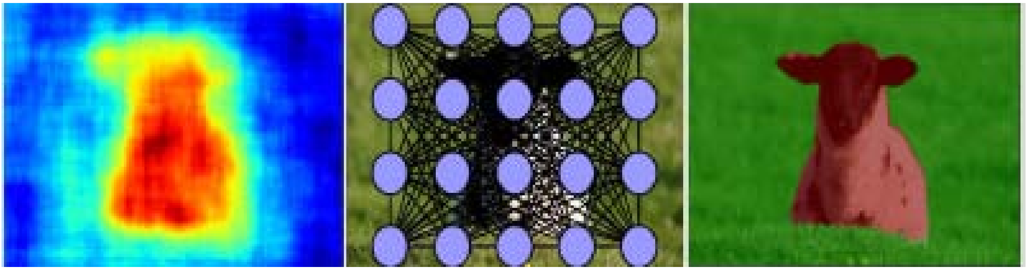
$\mathbf{X}$: Labels of the image. x_i: label of the ith pixel.

In CRF, we do not consider P($\mathbf{I}$), which is the probability of an image $\mathbf{I}$.

Simple implementation: use a feature extraction approach, e.g. CNN to obtain the features of $\mathbf{I}$, then use a classifier with softmax to obtain $P(\mathbf{X}|\mathbf{I})$, where elements in $\mathbf{X}$ are not correlated to each other.

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image



$$P(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})}\exp\left(-\boxed{\sum_{c\in\mathcal{C}_{\mathcal{G}}}\phi_c(\mathbf{X}_c|\mathbf{I})}\right) = \frac{\exp(-\boxed{E(\mathbf{x}\,|\,\mathbf{I})})}{Z(\mathbf{I})}$$

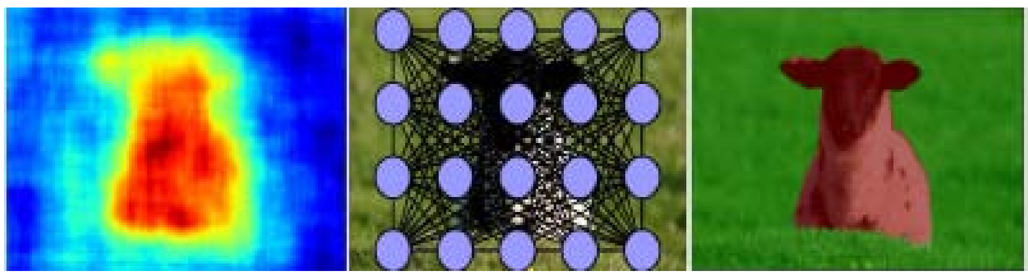
$$E(\mathbf{x}|\mathbf{I}) = \boxed{\sum_{c\in\mathcal{C}_{\mathcal{G}}}\phi_c(\mathbf{x}_c|\mathbf{I})}$$

$$E(\mathbf{x}) = \boxed{\sum_i \psi_u(x_i)} + \sum_{i<j}\psi_p(x_i, x_j),$$

Predictions made for each pixel independently (e.g. by an FCN)

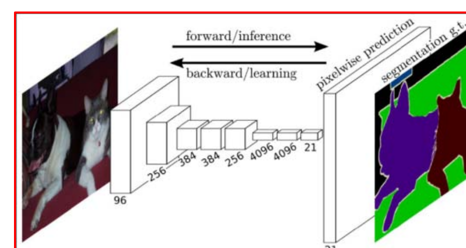
Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image



$$P(\mathbf{X}|\mathbf{I}) = \frac{1}{Z(\mathbf{I})}\exp\Big(-\boxed{\textstyle\sum_{c\in\mathcal{C}_{\mathcal{G}}}\phi_c(\mathbf{X}_c|\mathbf{I})}\Big) = \frac{\exp(-\boxed{E(\mathbf{x}\,|\,\mathbf{I})})}{Z(\mathbf{I})}$$

$$E(\mathbf{x}) = \boxed{\sum_i \psi_u(x_i)}$$



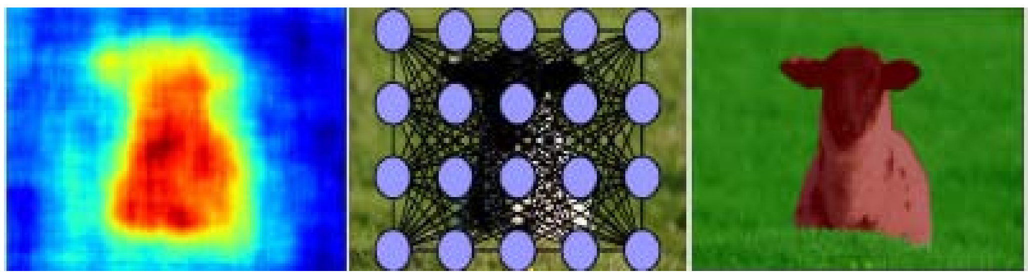
Predictions made for each pixel independently (e.g. by an FCN)

Special case:

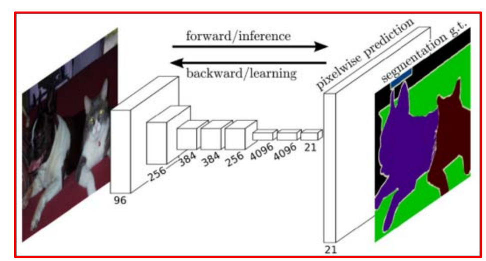
$$P(\mathrm{X}) = \prod_i \exp\left(-\sum_i w\mathrm{f}_i\right) / Z(\mathrm{I})$$

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image



$$E(\mathbf{x}) = \boxed{\sum_i \psi_u(x_i)}$$

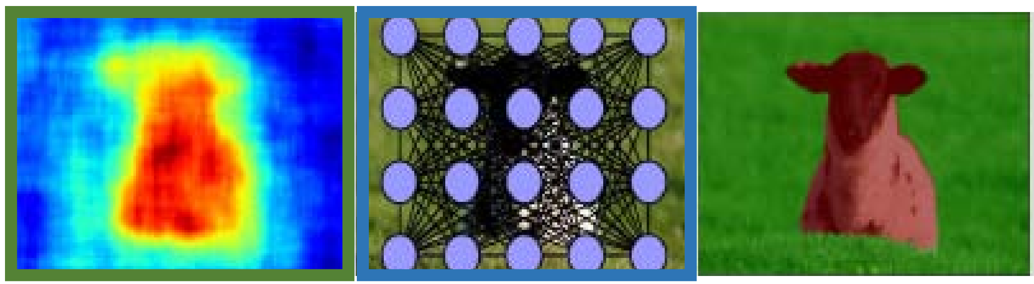


Predictions made for each pixel independently (e.g. by an FCN)

Special case: $P(\mathrm{X}) = \prod_i \exp\left(-\sum_i w\mathrm{f}_i\right) / Z(\mathrm{I})$ Softmax for multi-class classifier

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image



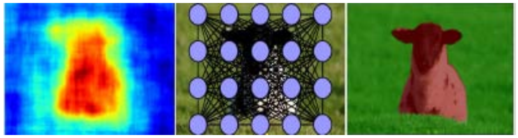
$$E(\mathbf{x}) = \boxed{\sum_i \psi_u(x_i)} + \boxed{\sum_{i<j} \psi_p(x_i, x_j)}$$

Predictions made for each pixel independently (e.g. by an FCN)

Penalizes similar pixels having different labels

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image



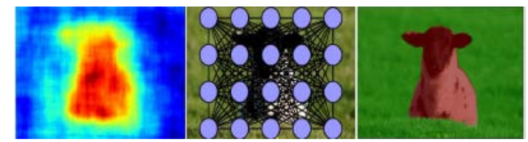
$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{k(\mathbf{f}_i, \mathbf{f}_j)},$$

$$E(\mathbf{x}) = \boxed{\sum_i \psi_u(x_i)} + \sum_{i<j} \boxed{\psi_p(x_i, x_j)},$$

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j),$$



$$\psi_p(x_i, x_j) = \boxed{\mu(x_i, x_j)} \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{\boxed{k(\mathbf{f}_i, \mathbf{f}_j)}},$$

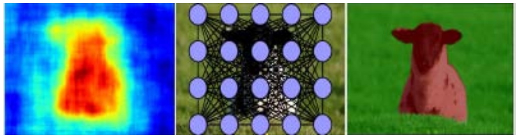
μ: label compatibility function

A simple label compatibility function is $\mu(x_i, x_j) = [x_i \neq x_j]$.
We can also learn the $\mu(x_i, x_j)$.

Conditional Random Fields (CRFs)

- The CRF accounts for contextual information in the image

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j).$$



$$\psi_p(x_i, x_j) = \boxed{\mu(x_i, x_j)} \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{\boxed{k(\mathbf{f}_i, \mathbf{f}_j)}},$$

μ: label compatibility function

$$\boxed{k(\mathbf{f}_i, \mathbf{f}_j)} = \underbrace{w^{(1)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right)}_{\text{appearance kernel}} + \underbrace{w^{(2)} \exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right)}_{\text{smoothness kernel}}$$

Conditional Random Fields (CRFs)

- $w^{(1)}=1$, $x_i \neq x_j$, $\mu(x_i, x_j) = 1$, same p_i-p_j
  - large $|I_i$-$I_j|$ -> large probability to have different labels
  - Small $|I_i$-$I_j|$ -> small probability to have different labels

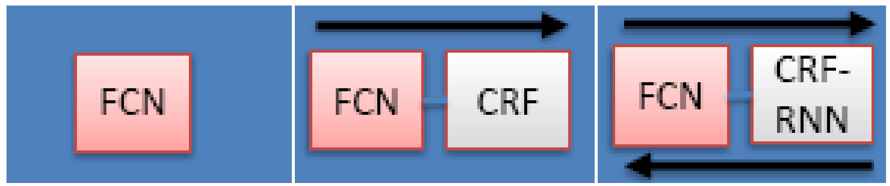
$$\psi_p(x_i, x_j) = \mu(x_i, x_j) \underbrace{\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{k(\mathbf{f}_i, \mathbf{f}_j)},$$

$$\mu(x_i, x_j) = [x_i \neq x_j]$$

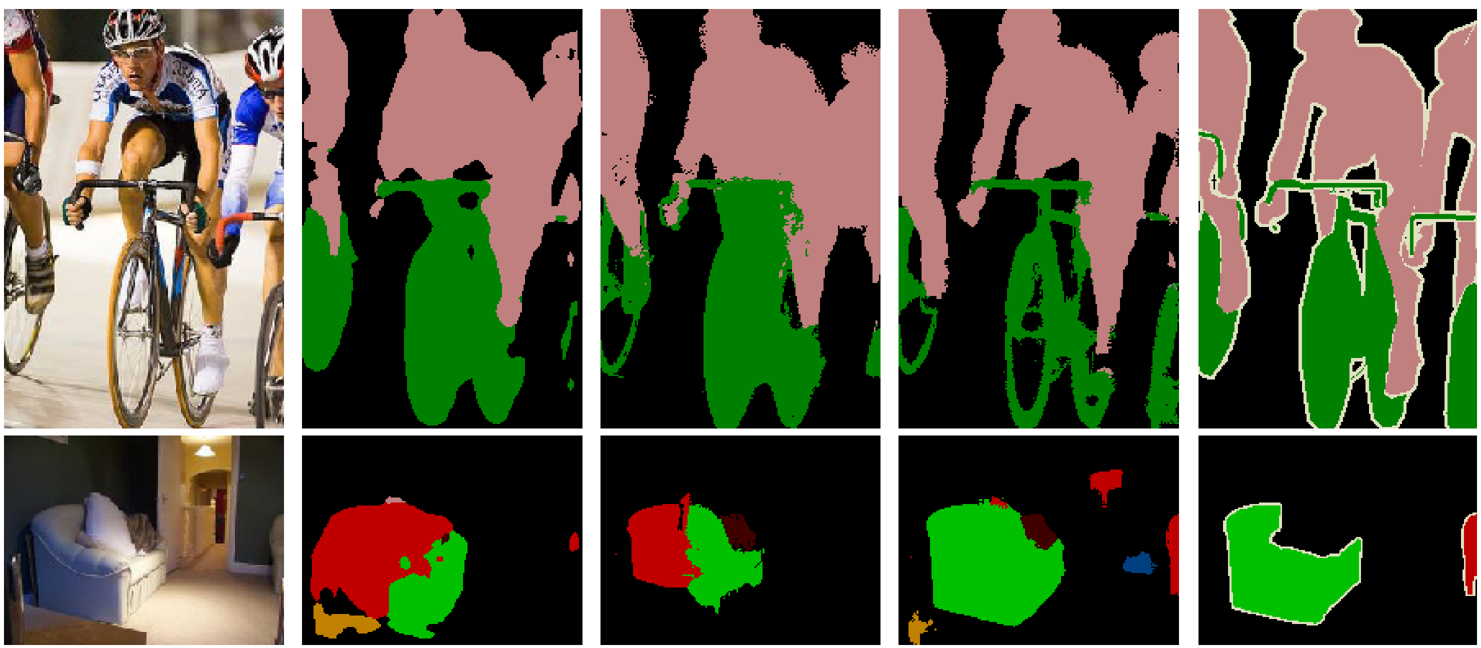
$$\boxed{k(\mathbf{f}_i, \mathbf{f}_j)} = w^{(1)} \underbrace{\exp\left(-\frac{|p_i - p_j|^2}{2\theta_\alpha^2} - \frac{|I_i - I_j|^2}{2\theta_\beta^2}\right)}_{\text{appearance kernel}} + w^{(2)} \underbrace{\exp\left(-\frac{|p_i - p_j|^2}{2\theta_\gamma^2}\right)}_{\text{smoothness kernel}}$$

$$p(x) = \frac{\exp(-E(\mathbf{x}))}{Z} \qquad E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j).$$

Fully Connected CRFs as a CNN



[Long et al. CVPR 2015] | [Chen et al. ICLR 2015] | [S. Zheng et al. ICCV 2015] | Ground truth



Conditional Random Fields (CRFs)

- Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials
- [Krähenbühl& Koltun, NIPS 2011]

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

while not converged **do**

  $\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ 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

  $Q_i \leftarrow \frac{1}{Z_i} \exp\left(\breve{Q}_i(l)\right)$ ▷ Normalizing

end while

Conditional Random Fields (CRFs)

$$P(\mathrm{X} \mid \mathrm{I}) = \frac{\exp(-E(\mathrm{X}))}{Z} \Rightarrow P(\mathrm{X}) = \frac{\exp(-E(\mathrm{X}))}{Z}$$

$$E(\mathbf{x}) = \sum_i \psi_u(x_i) + \sum_{i<j} \psi_p(x_i, x_j), \qquad \psi_p(x_i, x_j) = \mu(x_i, x_j) \underbrace{\textstyle\sum_{m=1}^{K} w^{(m)} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j)}_{k(\mathbf{f}_i, \mathbf{f}_j)},$$

Mean field approximation:

Instead of computing the exact distribution $P(\mathbf{X})$, the mean field approximation computes a distribution $Q(\mathbf{X})$ that minimizes the KL-divergence $\mathbf{D}(Q||P)$ among all distributions Q that can be expressed as a product of independent marginals $Q(\mathbf{X}) = \prod_i Q_i(X_i)$

$$Q_i(x_i = l) = \frac{1}{Z_i} \exp\left\{ -\psi_u(x_i) - \sum_{l' \in \mathcal{L}} \mu(l, l') \sum_{m=1}^{K} w^{(m)} \sum_{j \neq i} k^{(m)}(\mathrm{f}_i, \mathrm{f}_j) Q_j(l') \right\}$$

Conditional Random Fields (CRFs)

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

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ 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

$Q_i \leftarrow \frac{1}{Z_i} \exp\left(\breve{Q}_i(l)\right)$

▷ Normalizing

end while

$$Q_i(x_i = l) = \frac{1}{Z_i} \exp \left\{ -\psi_u(x_i) \quad \right\}$$

$$U_i(l) = -\psi_u(X_i = l)$$

Conditional random fields as recurrent neural networks

S. Zheng et al. ICCV 2015

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

while not converged **do**

$\tilde{Q}_i^{(m)}(l) \leftarrow \sum_{j \neq i} k^{(m)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$ 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

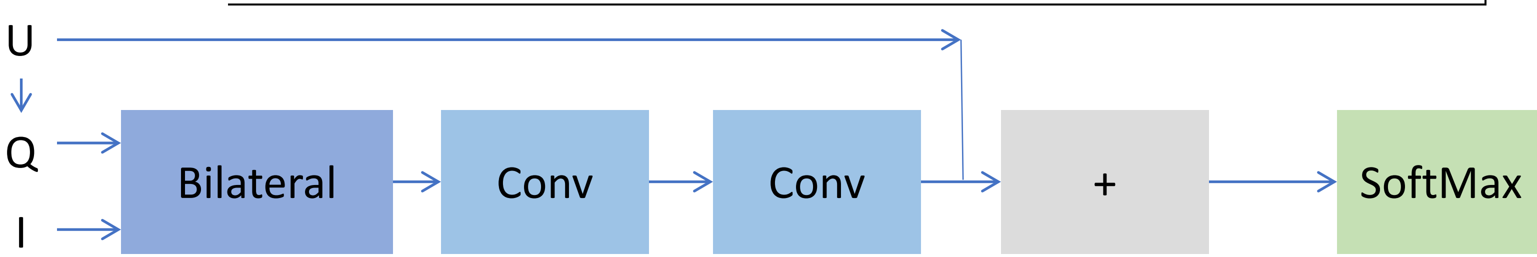
$\breve{Q}_i(l) \leftarrow U_i(l) - \hat{Q}_i(l)$

▷ Adding Unary Potentials

$Q_i \leftarrow \frac{1}{Z_i} \exp\left(\breve{Q}_i(l)\right)$

▷ Normalizing

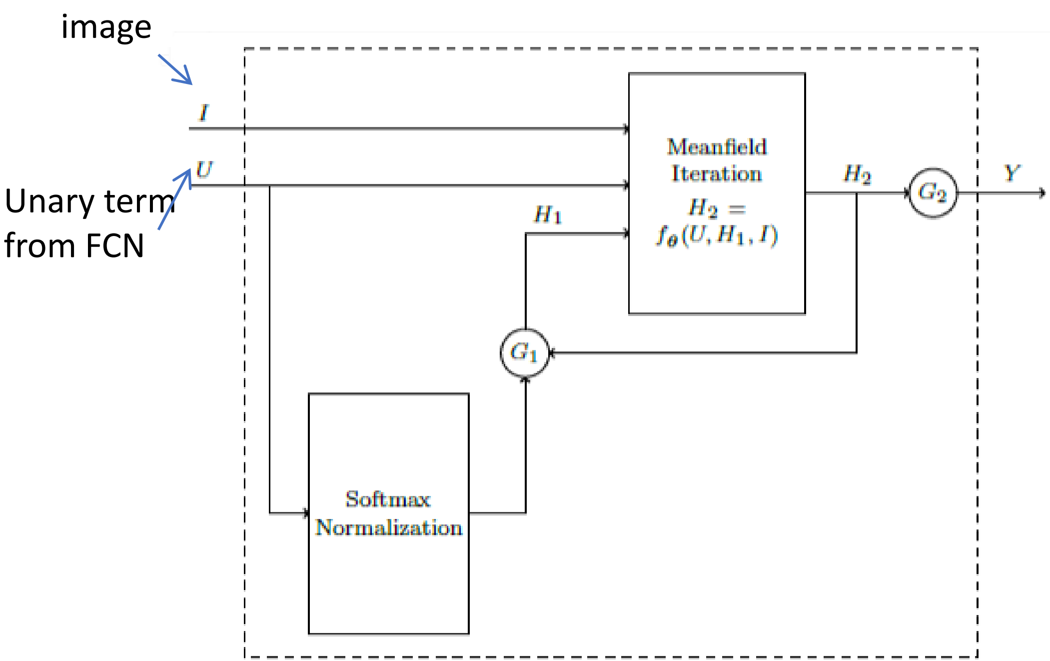
end while



Conditional random fields as recurrent neural networks

S. Zheng et al. ICCV 2015

CRF as RNN for multiple iterations



$$H_1(t) = \begin{cases} \text{softmax}(U), & t = 0 \\ H_2(t-1), & 0 < t \leq T, \end{cases}$$

$$H_2(t) = f_{\boldsymbol{\theta}}(U, H_1(t), I), \quad 0 \leq t \leq T,$$

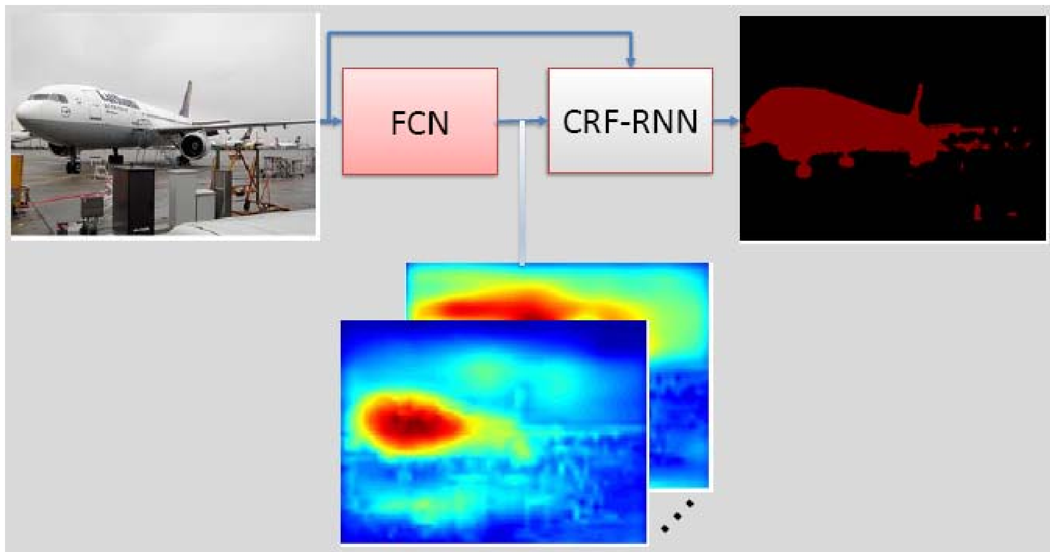
$$Y(t) = \begin{cases} 0, & 0 \leq t < T \\ H_2(t), & t = T. \end{cases}$$

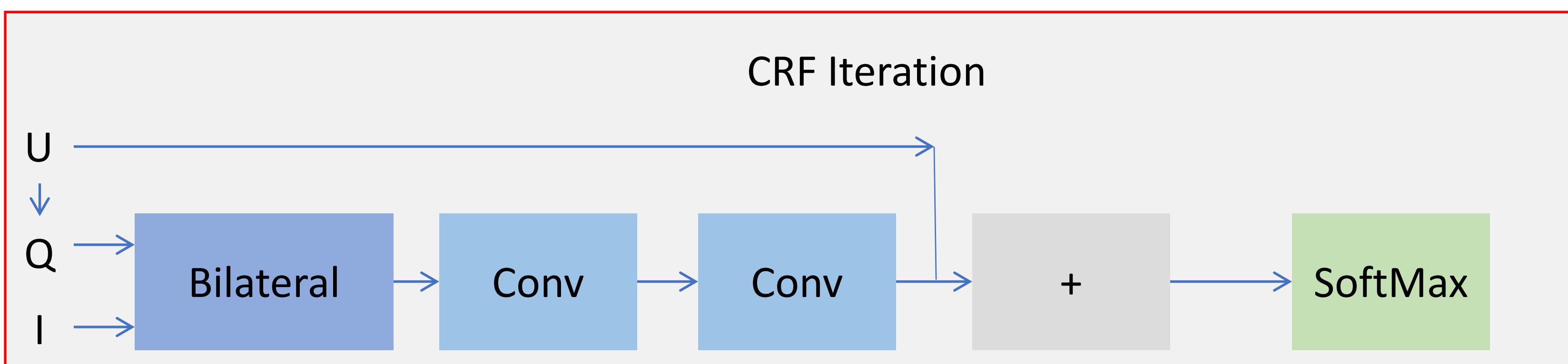
Total number of iterations

- $H2(t)$ is the output of the mean-field iteration $f\theta(U,H1(t),I)$.

Fully Connected CRFs as a CNN

- Putting together





Conditional random fields as recurrent neural networks

S. Zheng et al. ICCV 2015

PASCAL VOC

Method	Without COCO	With COCO
Plain FCN-8s	61.3	68.3
FCN-8s and CRF disconnected	63.7	69.5
End-to-end training of CRF-RNN	69.6	72.9

Mean IU Accuracy on PASCAL VOC 2012 Validation Set