max planck institut informatik

SIC Saarland Informatics Campus

Probabilistic Graphical Models and Their Applications

Dense Conditional Random Fields for Semantic Image Segmentation

@ Jan 6, 2021

Bernt Schiele

www.mpi-inf.mpg.de/gm/

Max Planck Institute for Informatics & Saarland University, Saarland Informatics Campus Saarbrücken

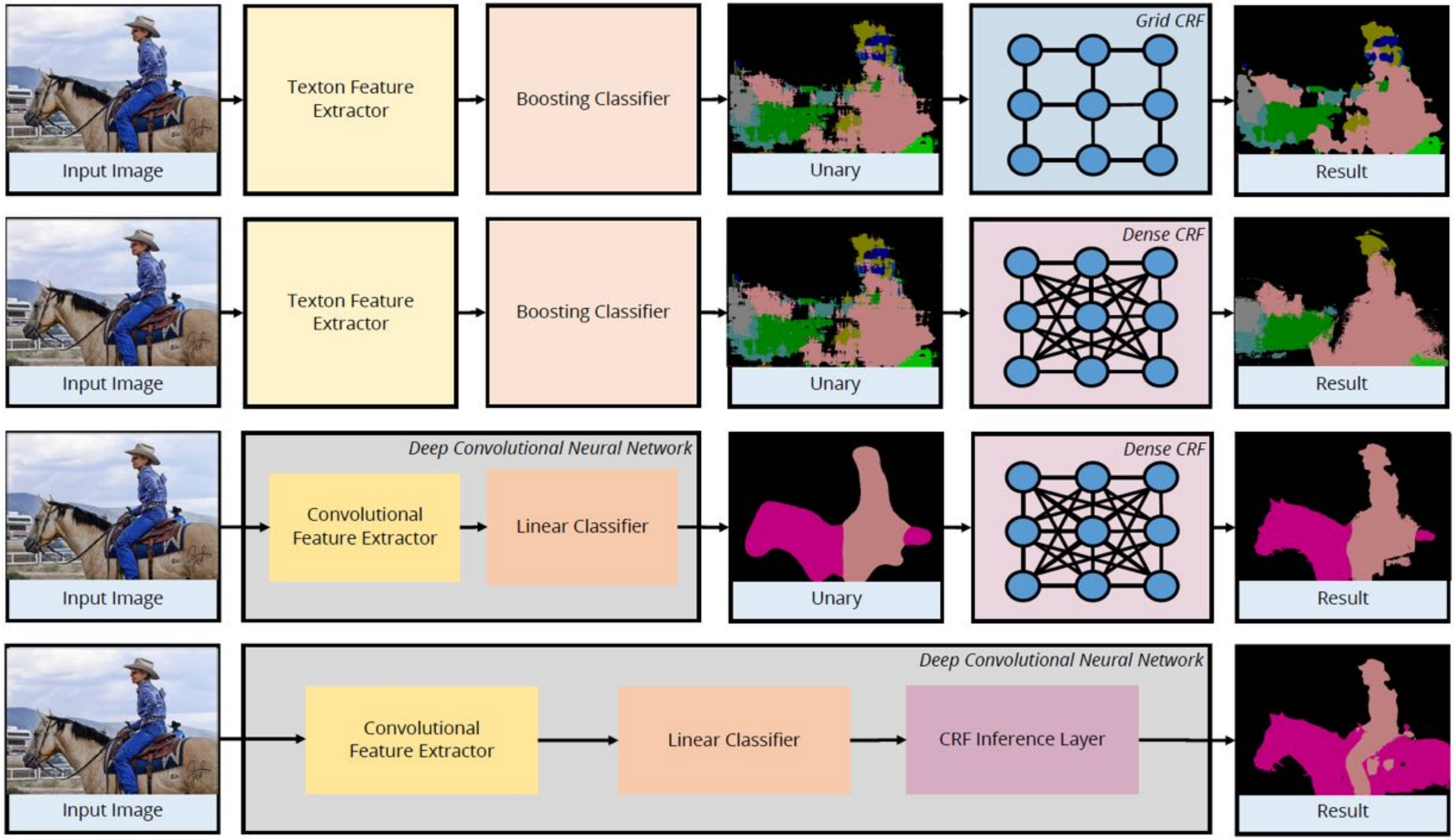
Overview Today's Lecture

- Semantic Image Segmentation as a Dense Labeling Problem
- Conditional Random Field (CRF) Models
  - vs. Markov Random Field Models
- Dense CRF Model
- Integration of Deep Learning and CRFs

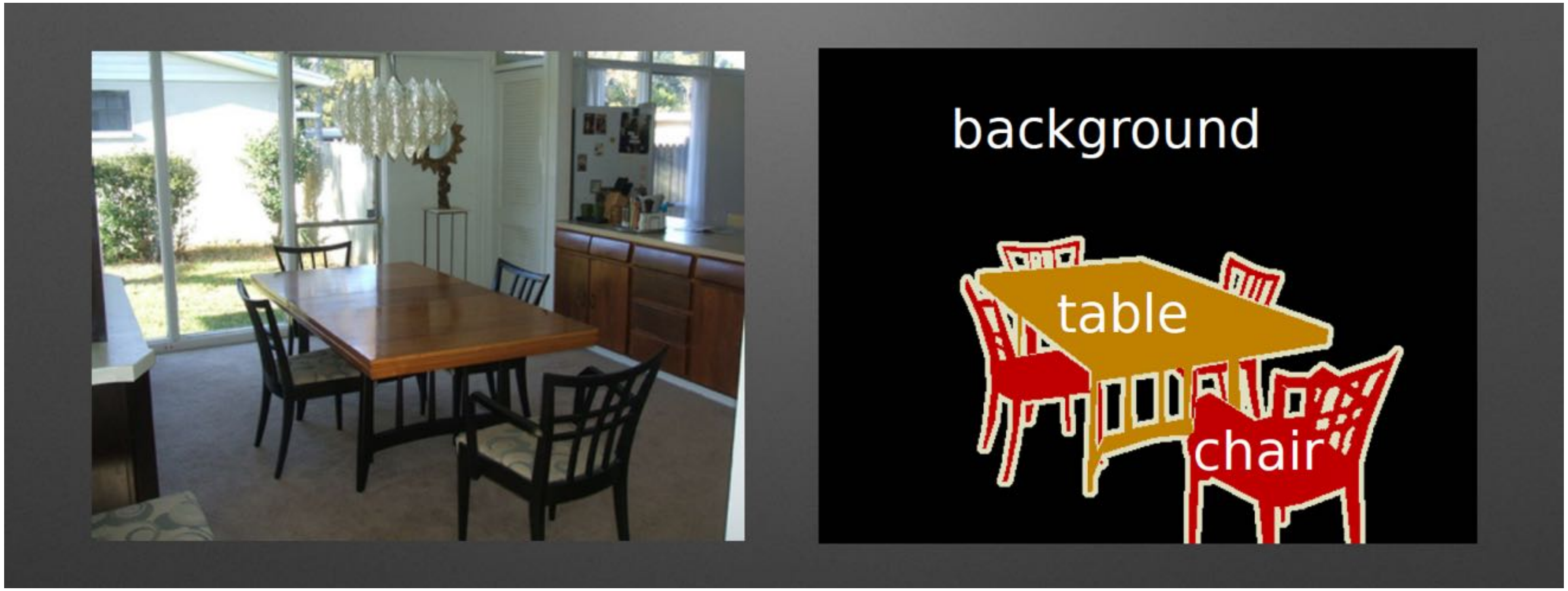
- Suggested Readings:
  - [1] Efficient Inference in Fully Contected CRFs with Gaussian Edge Potential, Philipp Krähenbühl and Vladlen Koltun, NeurIPS 2011 (https://arxiv.org/abs/1210.5644)
  - [2] Conditional Random Fields Meet Deep Neural Networks for Semantic Segmentation, Arnab, Zheng, et al., IEEE Sig. Proc. Magazine, 2018 (https://www.robots.ox.ac.uk/~tvg/publications/2017/CRFMeetCNN4SemanticSegmentation.pdf)

Pictorial Overview of Today's Lecture

image credit: paper [2]



Semantic Image Segmentation

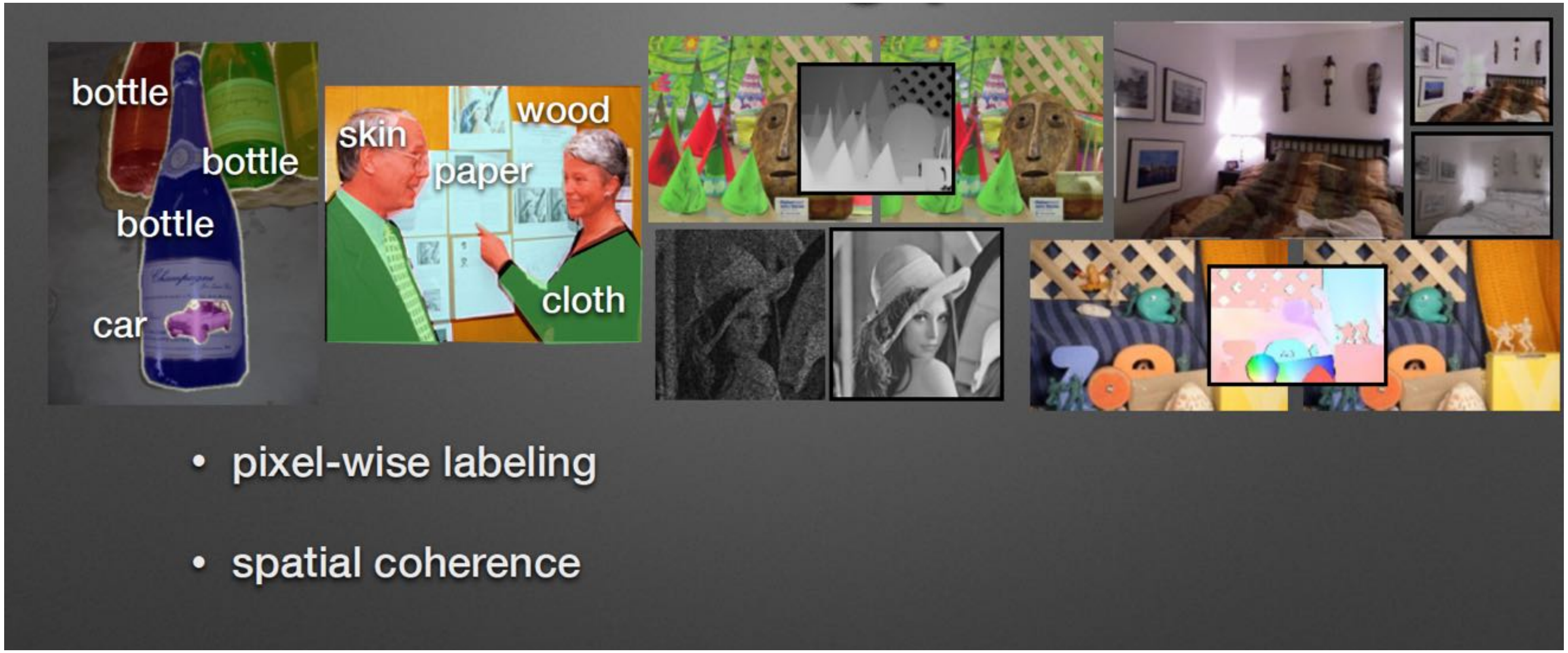


slide credit: Philipp Krähenbühl

Semantic Image Segmentation: Pixel-wise vs. Instance-Level

slide credit: Philipp Krähenbühl

Dense Labeling Problems

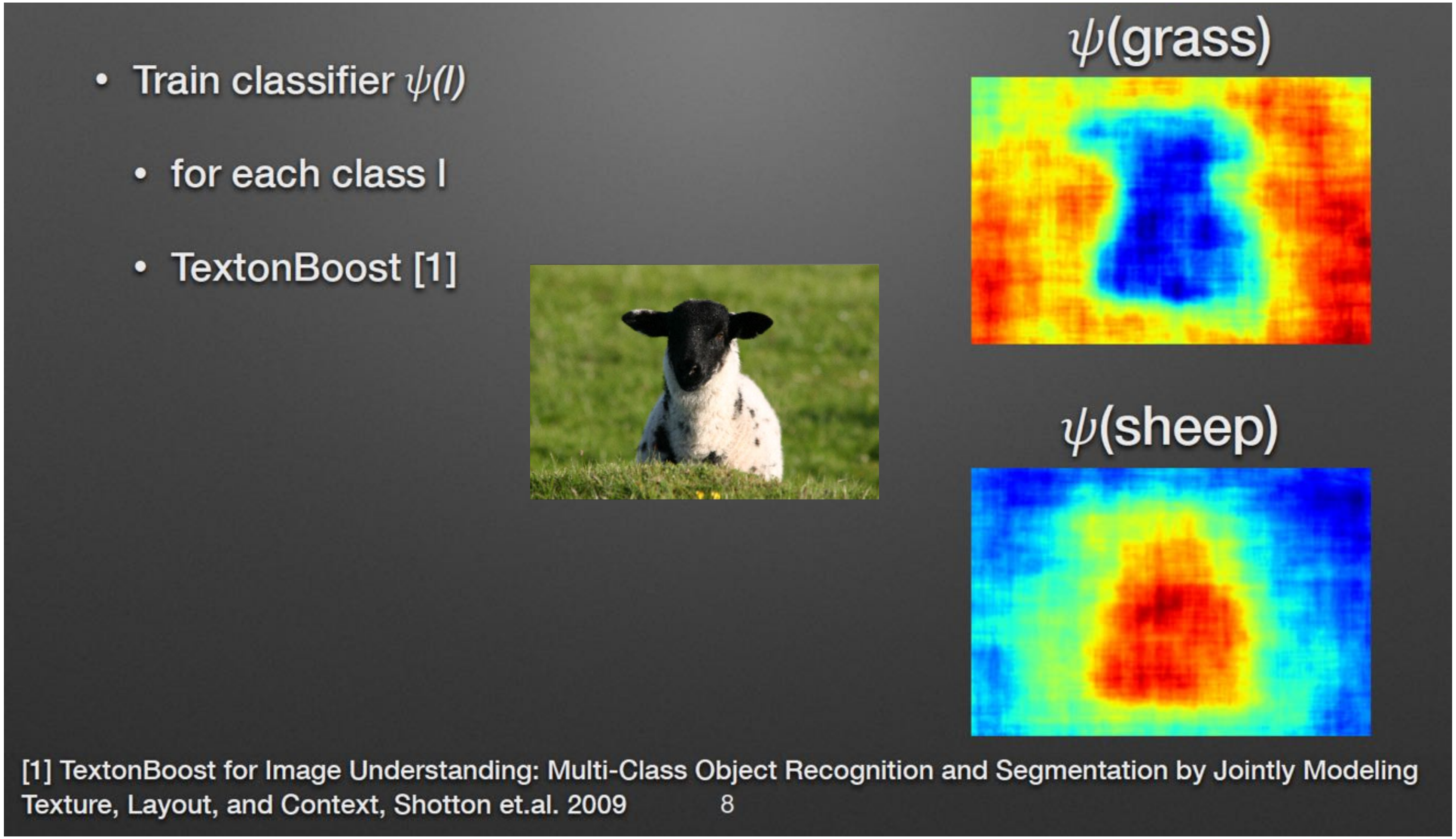


slide credit: Philipp Krähenbühl

Semantic Image Segmentation (Pixel-wise)

slide credit: Philipp Krähenbühl

Classification



Classification

- Train classifier $\psi(l)$
  - for each class l
  - TextonBoost [1]
- Pixels independent
  - noisy classification

[1] TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context, Shotton et.al. 2009 8

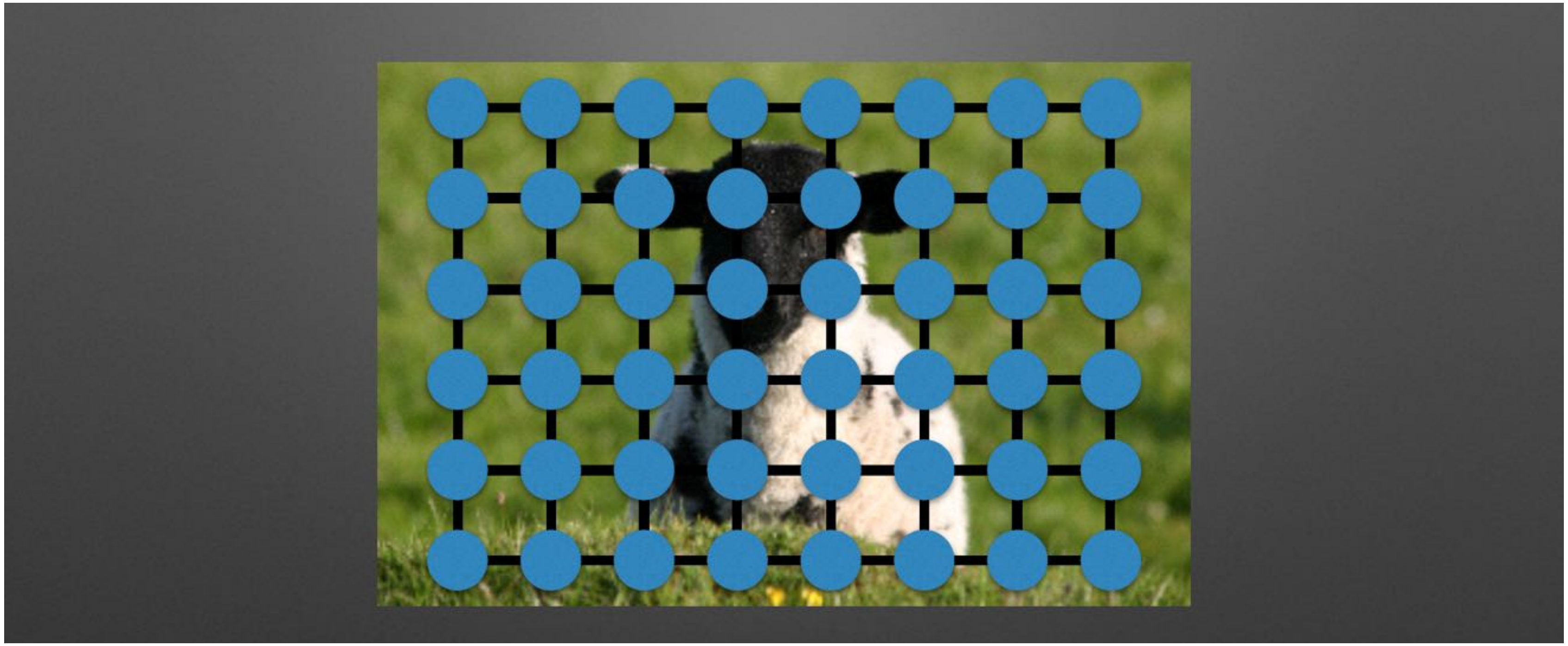
slide credit: Philipp Krähenbühl

Classification

- Train classifier $\psi(l)$
  - for each class l
  - TextonBoost [1]
- Pixels independent
  - noisy classification
- Large regional context
  - inaccurate around boundaries

[1] TextonBoost for Image Understanding: Multi-Class Object Recognition and Segmentation by Jointly Modeling Texture, Layout, and Context, Shotton et.al. 2009 8

Random Field Models



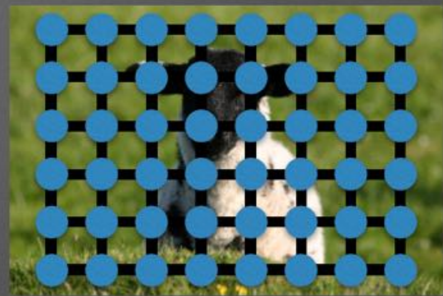
slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

unary term

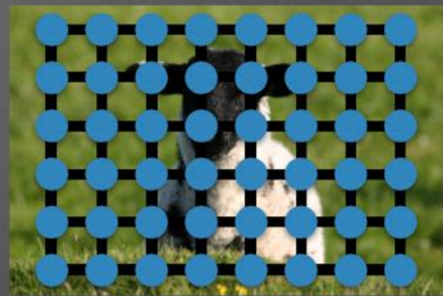
pairwise term



slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$



unary term

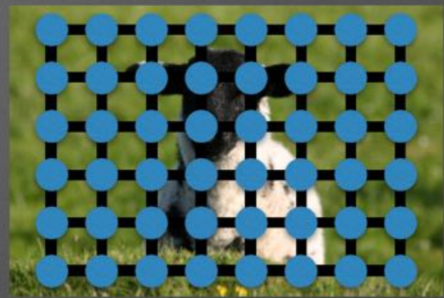
pairwise term

- Probabilistic interpretation $P(X) = \frac{1}{Z} \exp(-E(X))$

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$



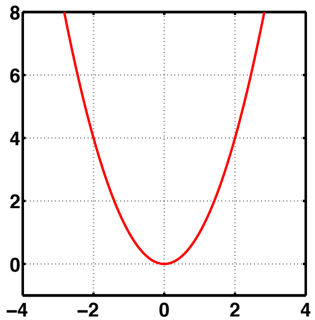
unary term

pairwise term

- Probabilistic interpretation $P(X) = \frac{1}{Z}\exp(-E(X))$
- MAP inference
  - most likely labeling
  - lowest energy

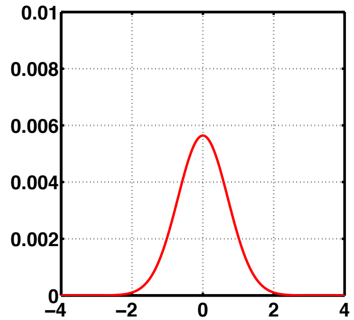
slide credit: Philipp Krähenbühl

Energy vs. Probability



Cost or energy

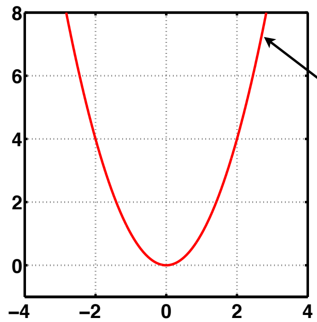
$$p(x) = \frac{1}{Z} e^{-E(x)} = \frac{1}{Z} e^{-x^2}$$



Probability density

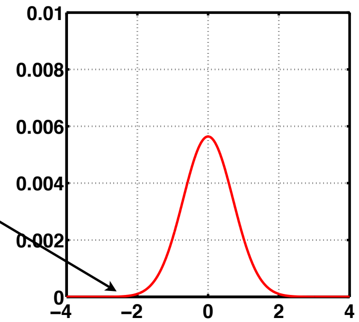
slide adapted from: Stefan Roth

Energy vs. Probability



Cost or energy

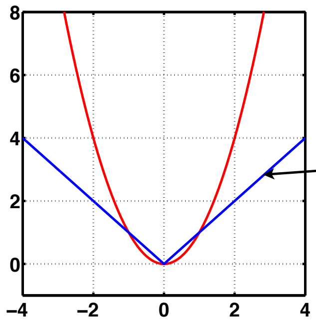
Large penalty / low probability for outliers



Probability density

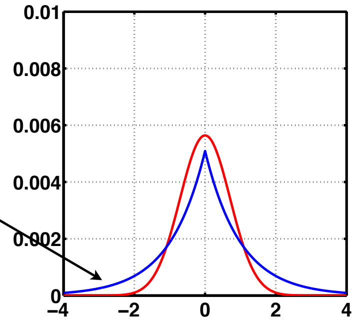
slide adapted from: Stefan Roth

Energy vs. Probability



Cost or energy

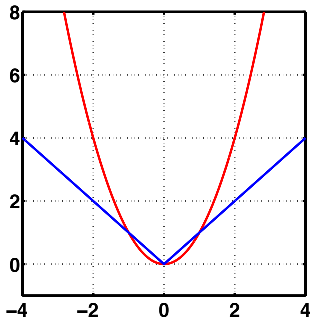
Smaller penalty / higher
probability for outliers



Probability density

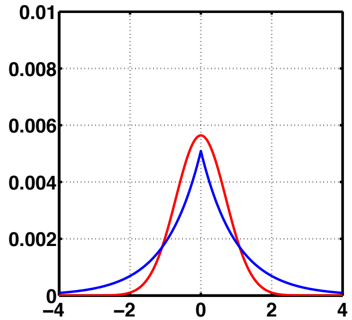
slide adapted from: Stefan Roth

Energy vs. Probability



Cost or energy

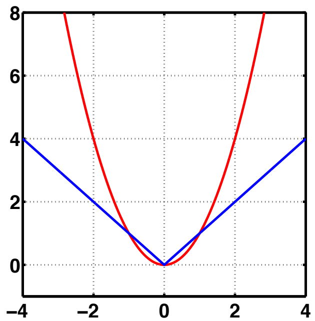
$$p(x) = \frac{1}{\hat{Z}} e^{-\hat{E}(x)} = \frac{1}{\hat{Z}} e^{-|x|}$$



Probability density

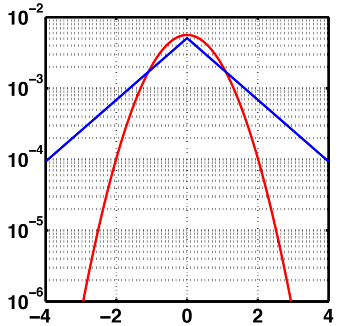
slide adapted from: Stefan Roth

Energy vs. Probability



Cost or energy

$$p(x) = \frac{1}{\hat{Z}} e^{-\hat{E}(x)} = \frac{1}{\hat{Z}} e^{-|x|}$$

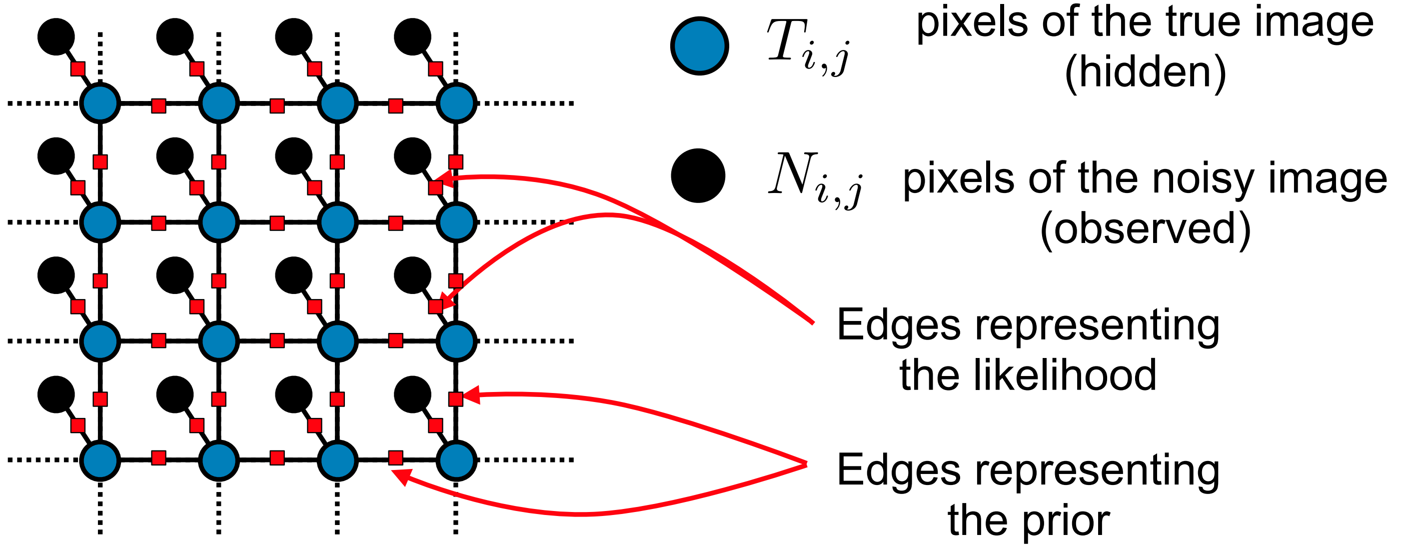


Probability density

slide adapted from: Stefan Roth

MRF Model of the (complete) Posterior for Image Denoising

- We can put the likelihood and the prior together in a single MRF model:



$$p(\mathbf{T}|\mathbf{N}) \propto p(\mathbf{N}|\mathbf{T})p(\mathbf{T})$$

$$= \left(\prod_{i,j} p(N_{i,j}|T_{i,j})\right)\left(\prod_{i,j} f_H(T_{i,j}, T_{i+1,j}) \cdot f_V(T_{i,j}, T_{i,j+1})\right)$$

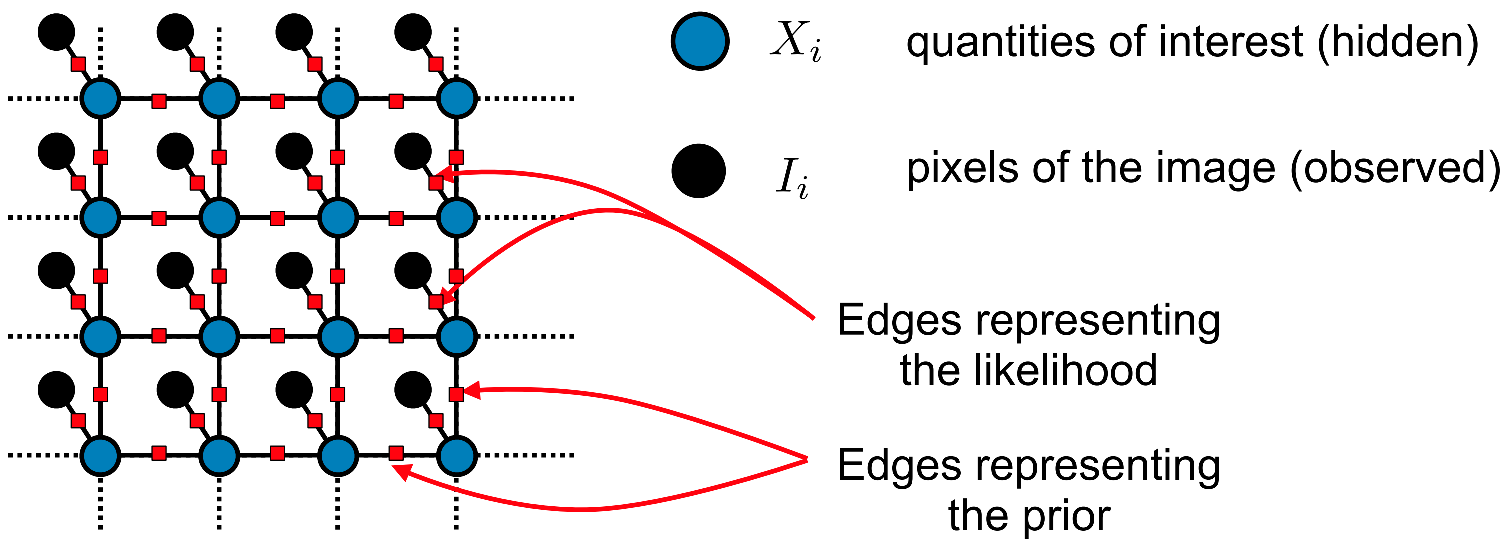
More Generally

$$p(X|I) = \frac{p(I|X)p(X)}{p(I)} \propto p(I|X)p(X)$$

- The quantity of interest: X = Output
  - true pixel values in image denoising
  - semantic labels in image segmentation

- The input / observation: I = Image
  - image denoising: I = noisy image
  - semantic segmentation: I = image

More Generally: Factorization Given the particular MRF Graph



$$p(X|I) \propto p(I|X)p(X) = \prod_i p(I_i|X_i) \prod_{i,j \in N_4} p(X_i, X_j)$$

slide adapted from: Stefan Roth

More Generally

- Goal of Inference often MAP (Maximum A Posteriori estimation):

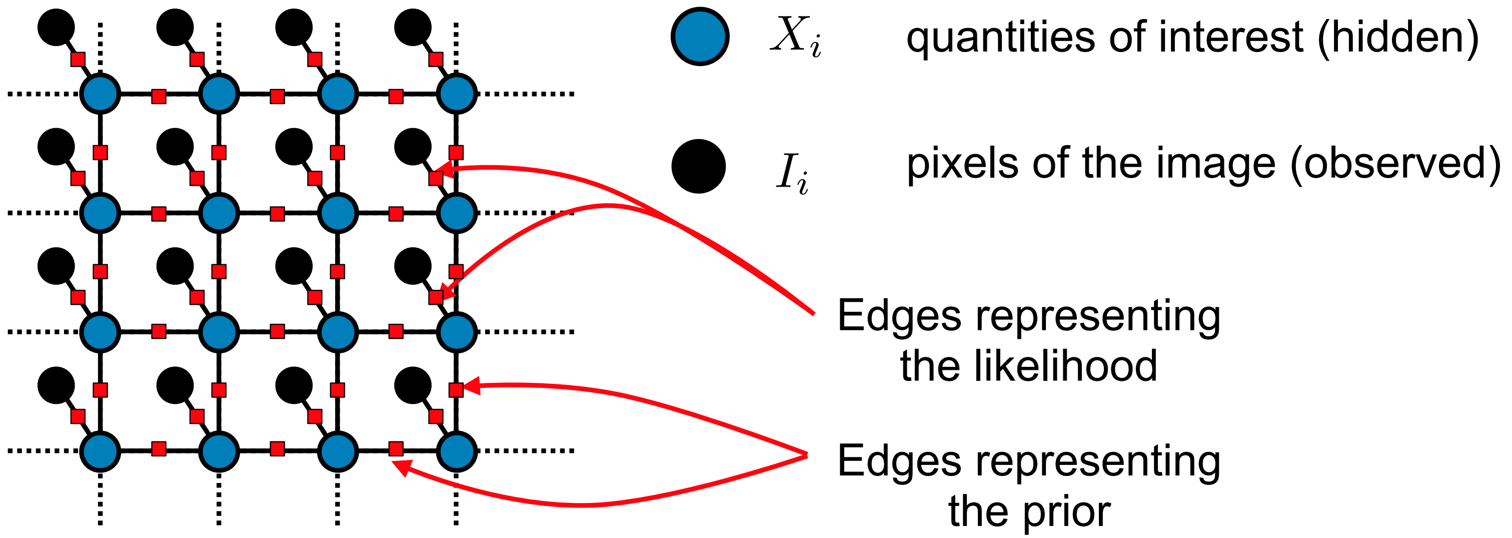
$$\begin{aligned}\arg\max_X p(X|I) &= \arg\max_X \left(p(I|X)p(X)\right) \\ &= \arg\min_X \left(-\log p(I|X) - \log p(X)\right)\end{aligned}$$

- For our MRF:
  - minimize the following "energy":

$$\begin{aligned}E(X) &= -\log p(I|X) - \log p(X) \\ &= -\sum_i \log p(I_i|X_i) - \sum_{i,j \in N_4} \log p(X_i, X_j) \\ &= \sum_i \psi_i(X_i|I) + \sum_{i,j \in N_4} \psi_{i,j}(X_i, X_j)\end{aligned}$$

unary terms   pairwise terms

More Generally: Factorization Given the particular MRF Graph

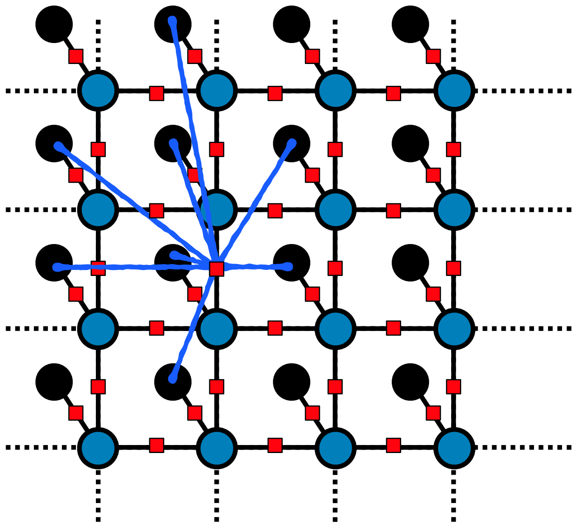


$$E(X) = \sum_i \psi_i(X_i|I) + \sum_{i,j \in N_4} \psi_{i,j}(X_i, X_j)$$

unary terms pairwise terms

slide adapted from: Stefan Roth

CRF (Conditional Random Field): Enhance Graphical Model with Additional Dependencies



- X_i quantities of interest (hidden)
- I_i pixels of the image (observed)

$$E(X) = \sum_i \psi_i(X_i|I) + \sum_{i,j \in N_4} \psi_{i,j}(X_i, X_j|I)$$

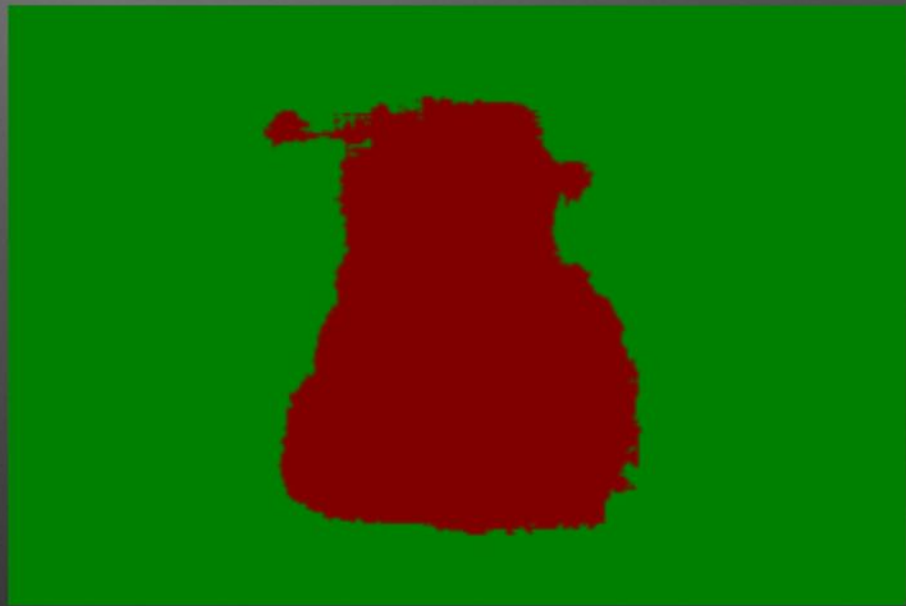
unary terms pairwise terms

slide adapted from: Stefan Roth

Random Field Models

$$E(X) = \sum_{i} \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$\psi_{ij}(X_i,X_j) = 0$



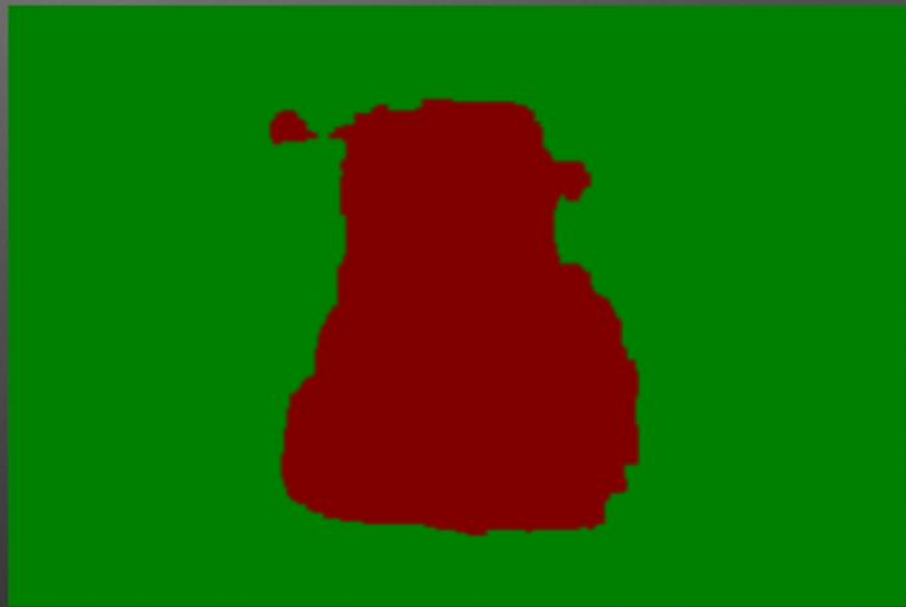
slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i,X_j) = [X_i \neq X_j]$$

Potts Model
(used e.g. for stereo matching



conditional random field

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i,X_j) = 100[X_i \neq X_j]$$

Potts Model
(used e.g. for stereo matching

conditional random field

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i,X_j) = 100[X_i \neq X_j]$$

Potts Model
(used e.g. for stereo matching

conditional random field

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i,X_j) = w_{ij}[X_i \neq X_j]$$

$$w_{ij} = \exp(-\alpha(c_i - c_j)^2)$$

conditional random field

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i,X_j) = w_{ij}[X_i \neq X_j]$$

$$w_{ij}=\exp(-\alpha(c_i-c_j)^2)$$

weight horizontal

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i, X_j) = w_{ij}[X_i \neq X_j]$$

$$w_{ij} = \exp(-\alpha(c_i - c_j)^2)$$

weight vertical

slide credit: Philipp Krähenbühl

Random Field Models

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in N} \psi_{ij}(X_i, X_j)$$

$\psi_{ij}(X_i,X_j) = 100w_{ij}[X_i \neq X_j]$

conditional random field
color sensitive

Random Field Models

Pros:

- Probabilistic interpretation
- Parameter learning
- Combine with other models

slide credit: Philipp Krähenbühl

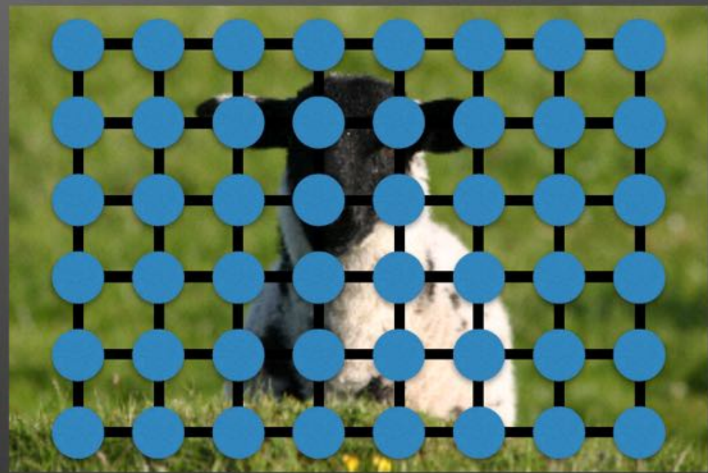
Random Field Models

Pros:

- Probabilistic interpretation
- Parameter learning
- Combine with other models

Cons:

- Shrinking bias
- Models only local interactions

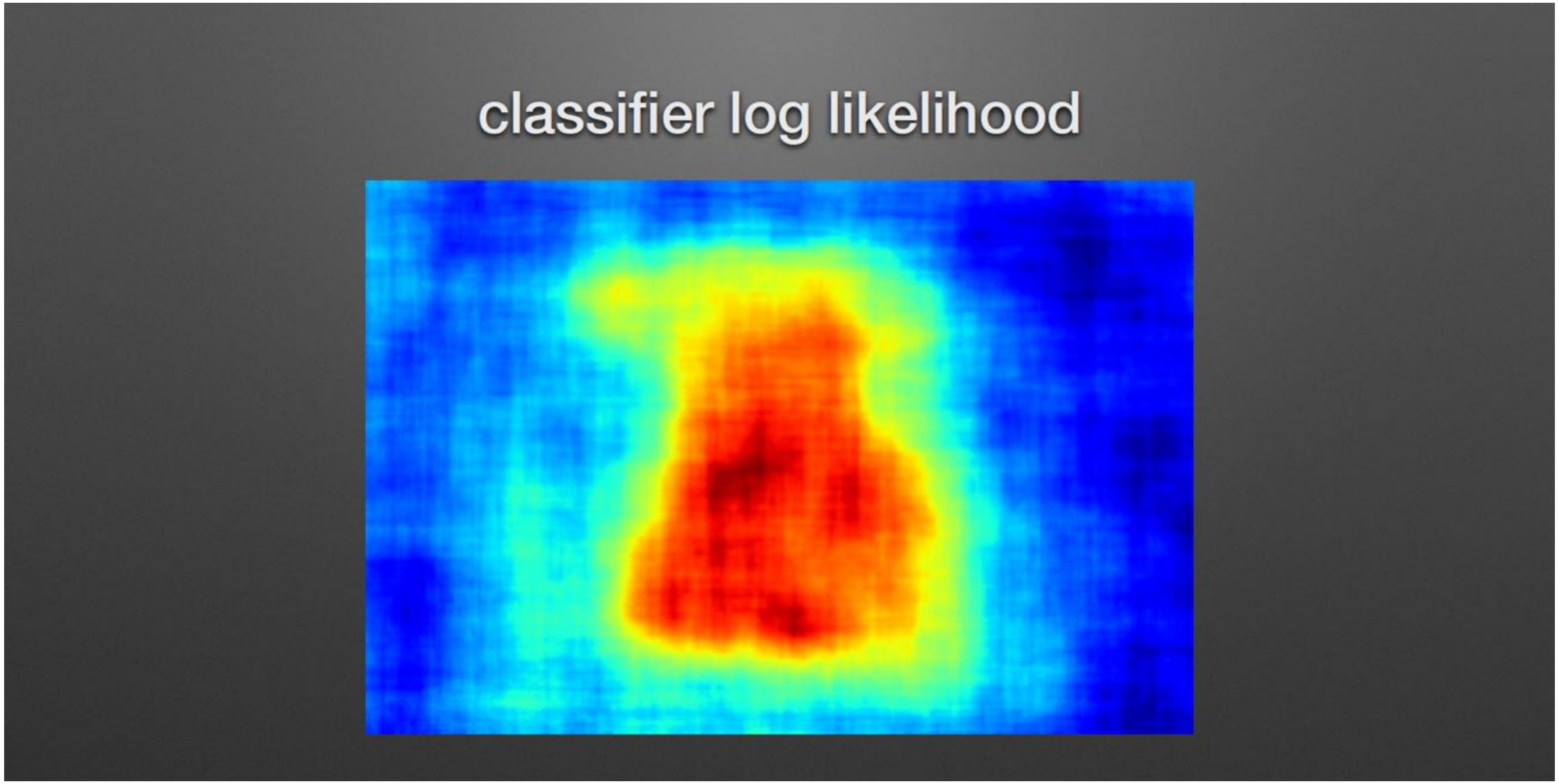


slide credit: Philipp Krähenbühl

Filtering

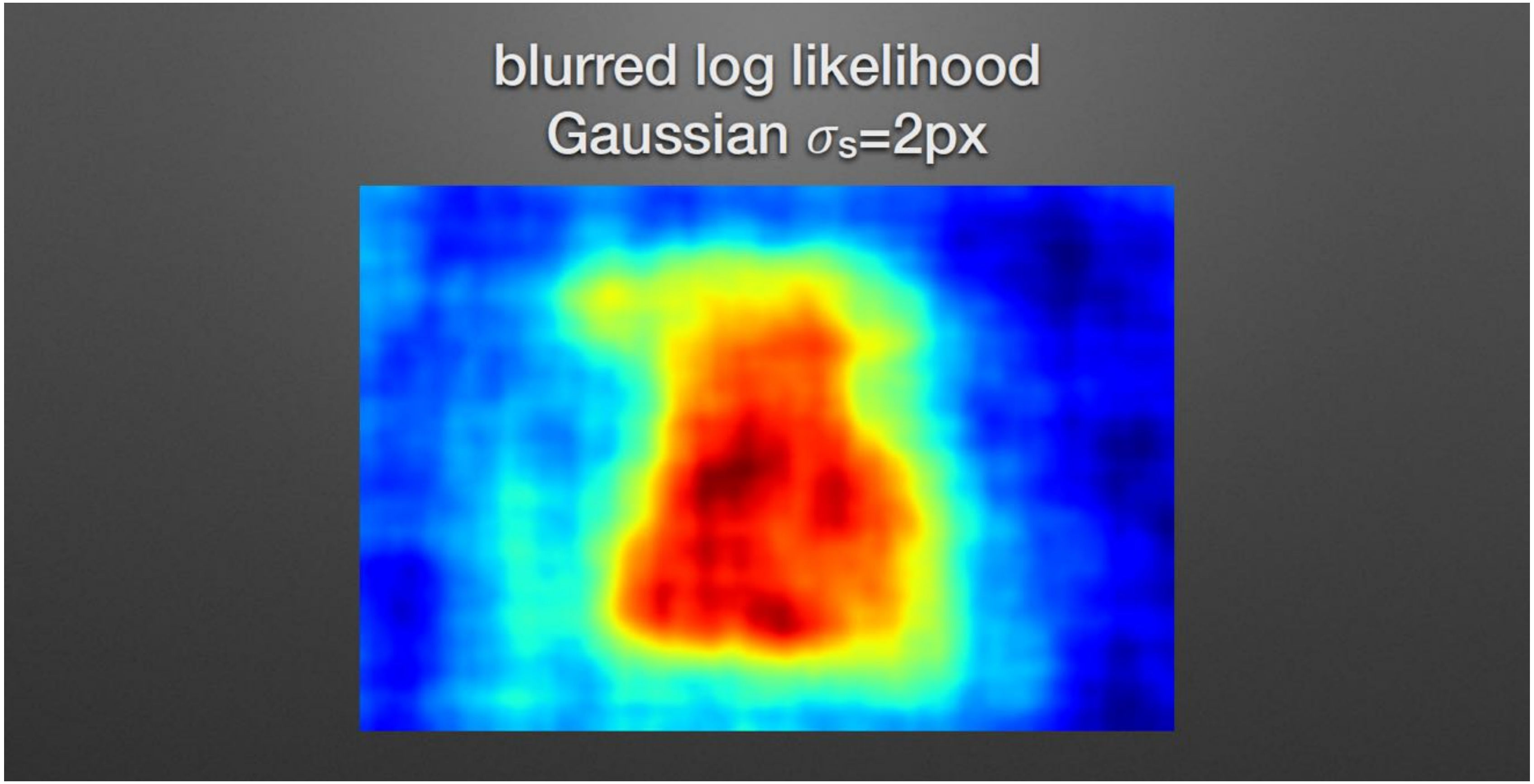
slide credit: Philipp Krähenbühl

Filtering



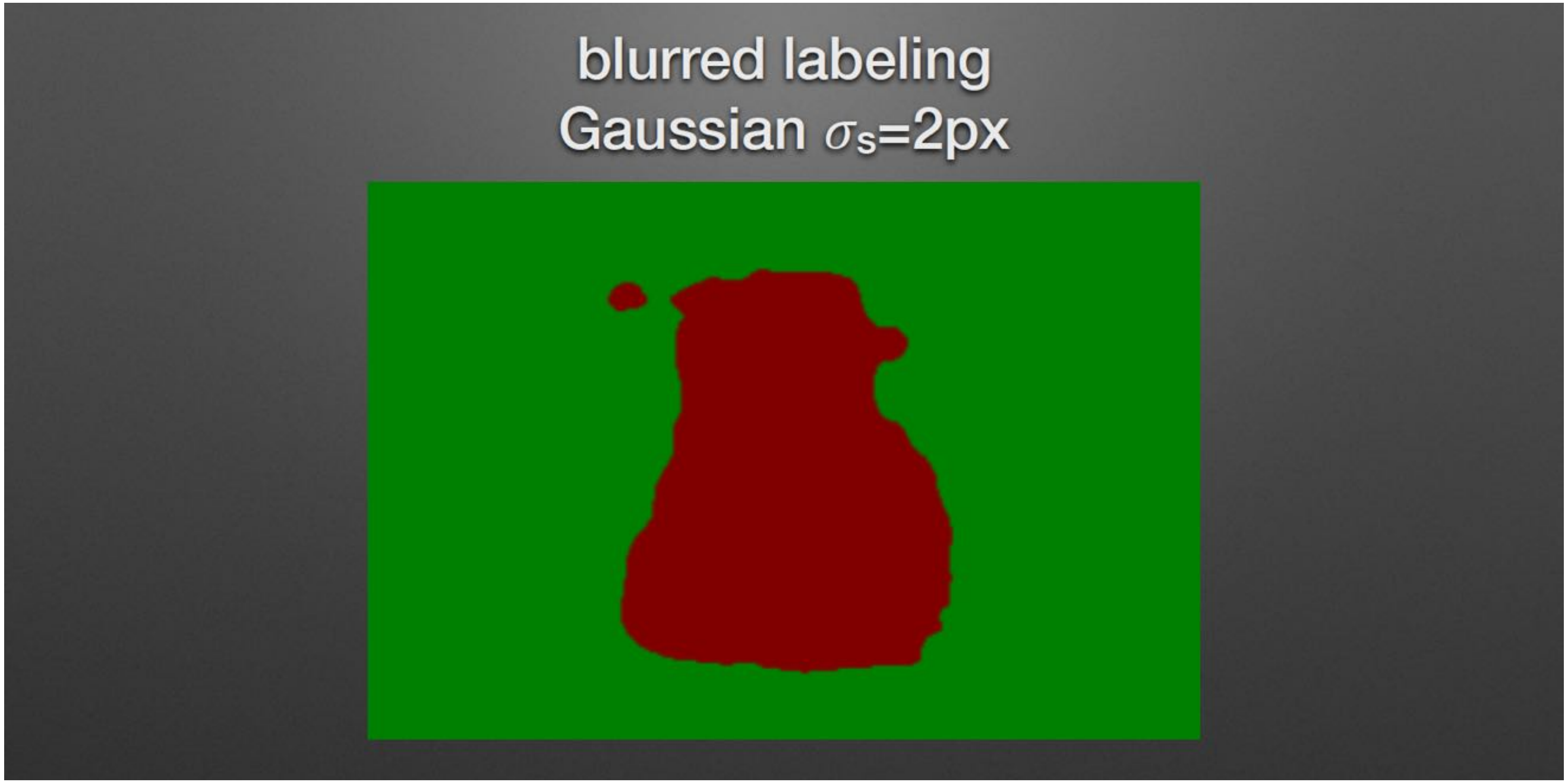
slide credit: Philipp Krähenbühl

Filtering



slide credit: Philipp Krähenbühl

Filtering



slide credit: Philipp Krähenbühl

Filtering

slide credit: Philipp Krähenbühl

Filtering

slide credit: Philipp Krähenbühl

Filtering

slide credit: Philipp Krähenbühl

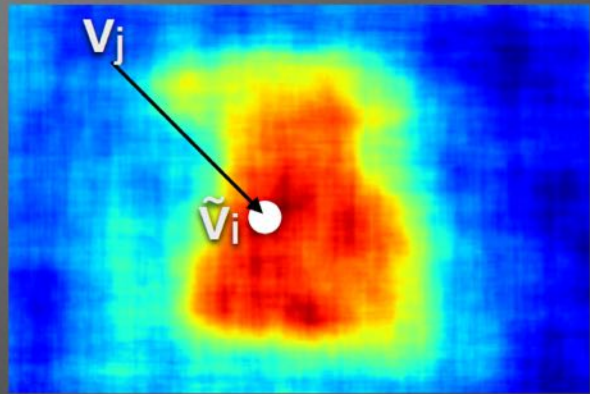
Filtering

slide credit: Philipp Krähenbühl

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$$w_{ij} = \exp(-(s_i - s_j)^2/\sigma_s)\ \exp(-(c_i - c_j)^2/\sigma_c)$$

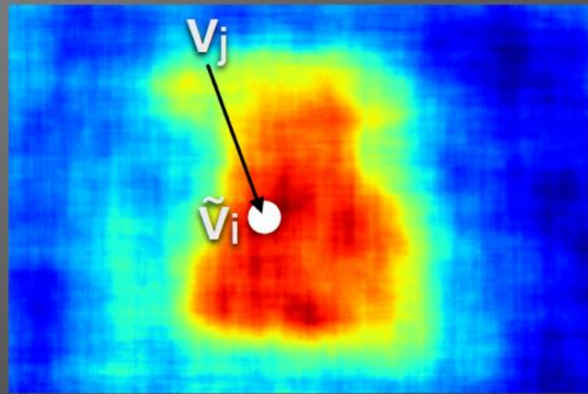


[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$$w_{ij} = \exp(-(s_i - s_j)^2/\sigma_s) \ \exp(-(c_i - c_j)^2/\sigma_c)$$

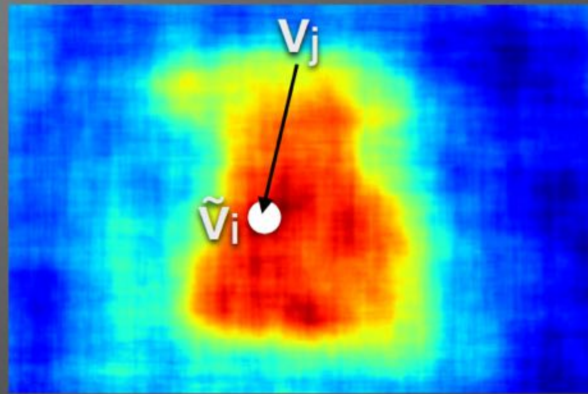


[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$$w_{ij} = \exp(-(s_i - s_j)^2/\sigma_s)\ \exp(-(c_i - c_j)^2/\sigma_c)$$

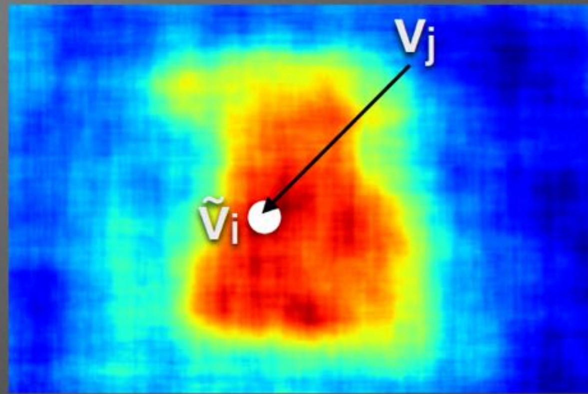


[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$$w_{ij} = \exp(-(s_i-s_j)^2/\sigma_s)\ \exp(-(c_i-c_j)^2/\sigma_c)$$

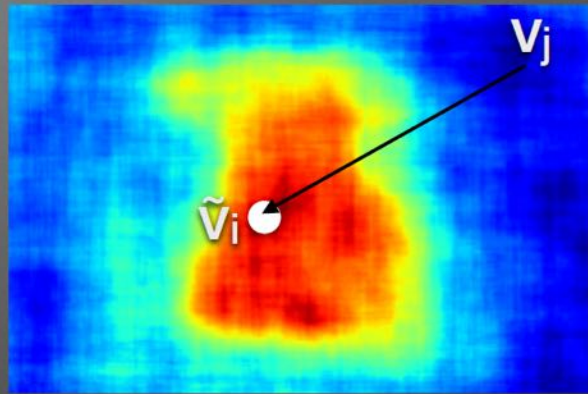


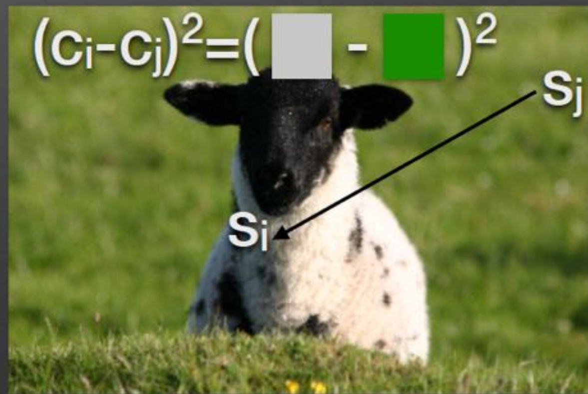
[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$w_{ij} = \exp(-(s_i - s_j)^2/\sigma_s)\ \exp(-(c_i - c_j)^2/\sigma_c)$



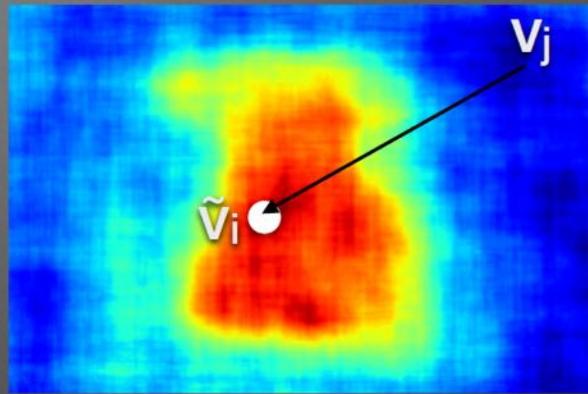


[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

w $\exp(-(s_i-s_j)^2/\sigma_s)$ exp

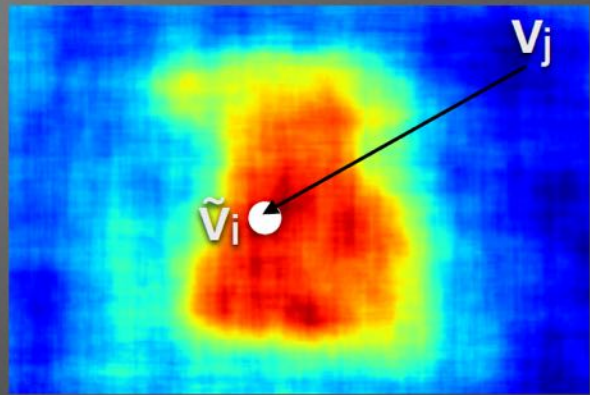


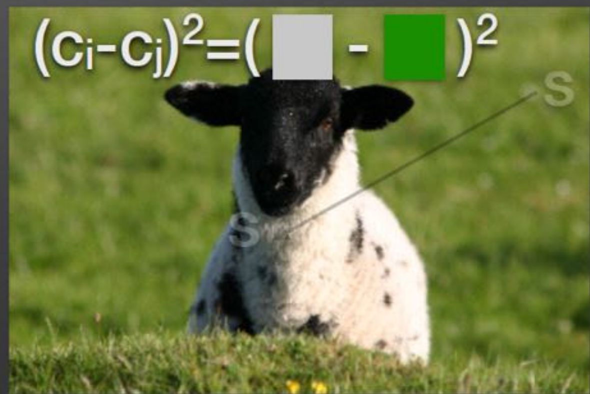
[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

w exp $\exp(-(c_i - c_j)^2/\sigma_c)$





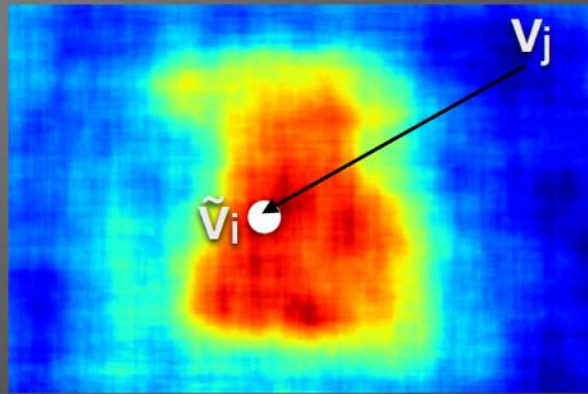
[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

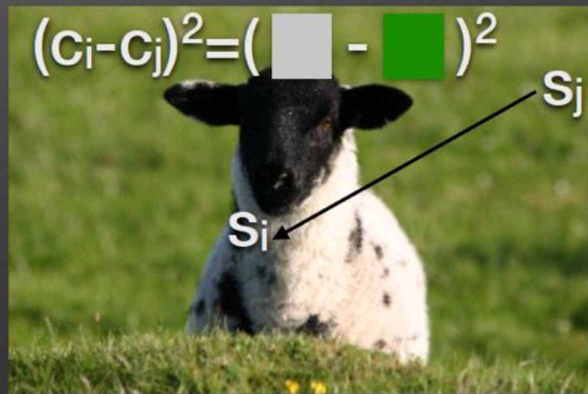
Filtering

$$\tilde{v}_i = \sum_j w_{ij} v_j$$

$w_{ij} = \exp(-(s_i-s_j)^2/\sigma_s)\ \exp(-(c_i-c_j)^2/\sigma_c)$

- Efficient convolution
  - Permutohedral lattice [2]
  - compute all $\tilde{v}_i$ in linear time
  - 50-100ms / image



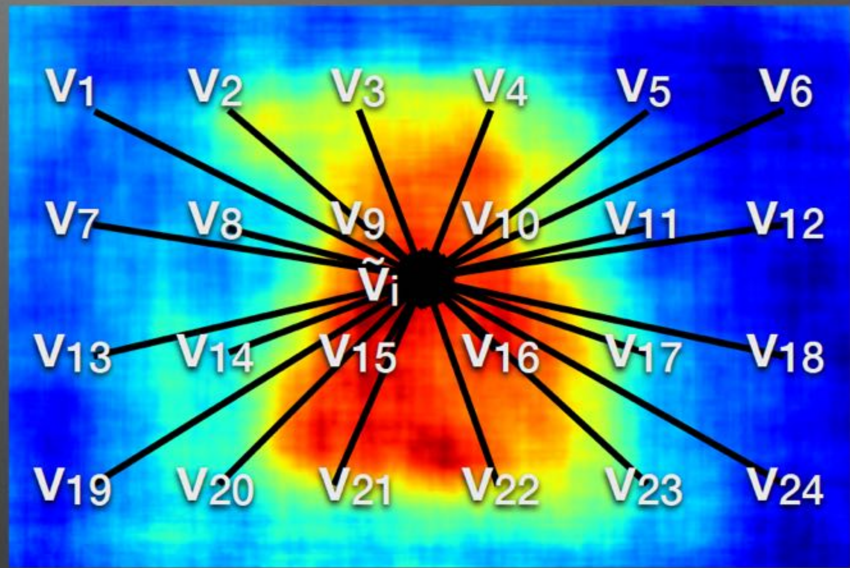


[2] Fast High-Dimensional Filtering Using the Permutohedral Lattice, Adams et.al. 2010

Filtering

Pros:

- Propagates information over large distances
  - up to 1/3 of image



slide credit: Philipp Krähenbühl

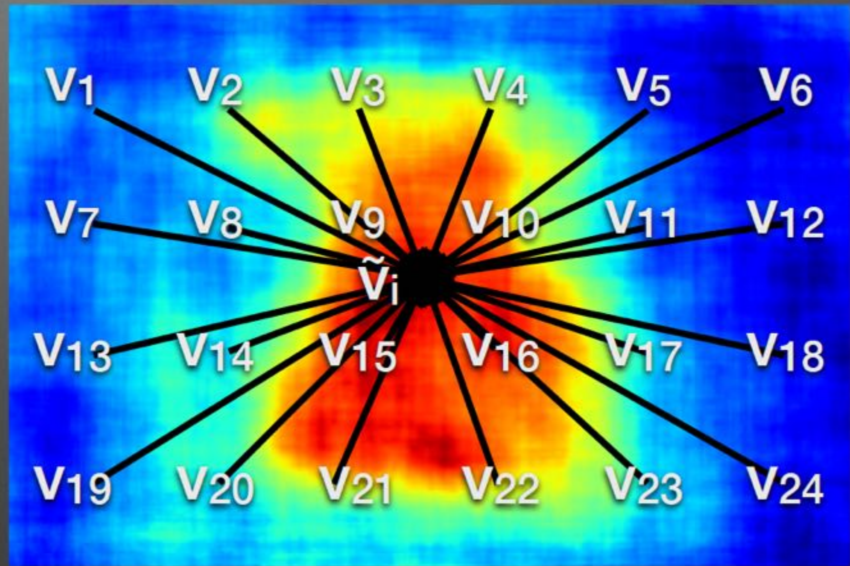
Filtering

Pros:

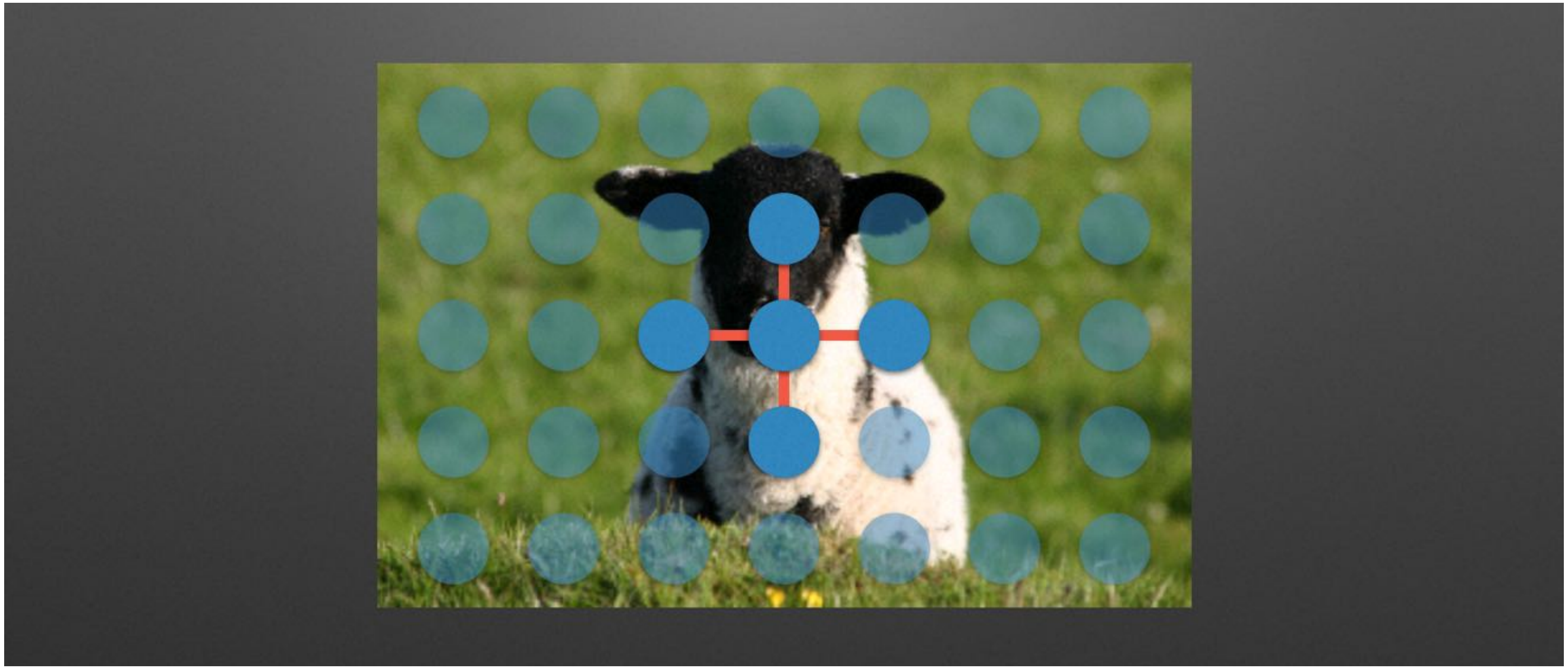
- Propagates information over large distances
  - up to 1/3 of image

Cons:

- No probabilistic interpretation
- No joint inference
- No learning

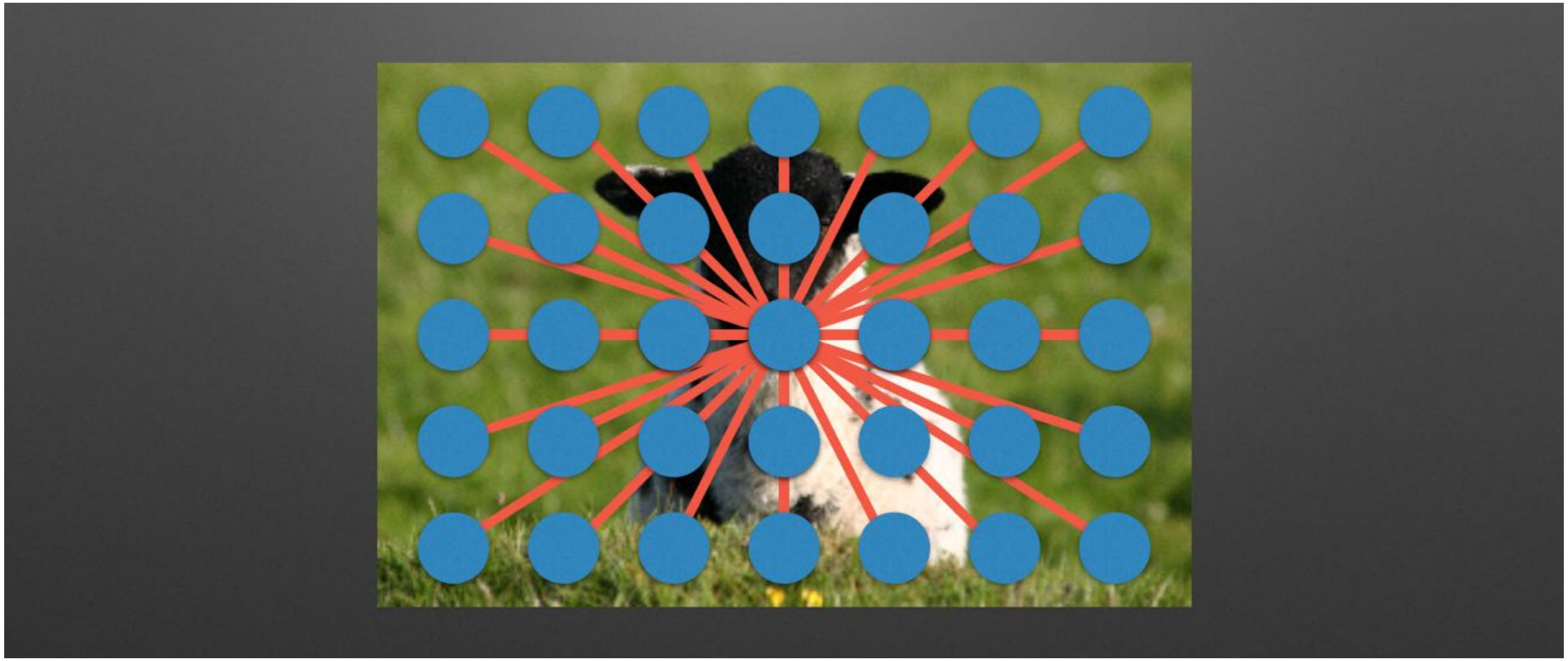


Dense Random Fields



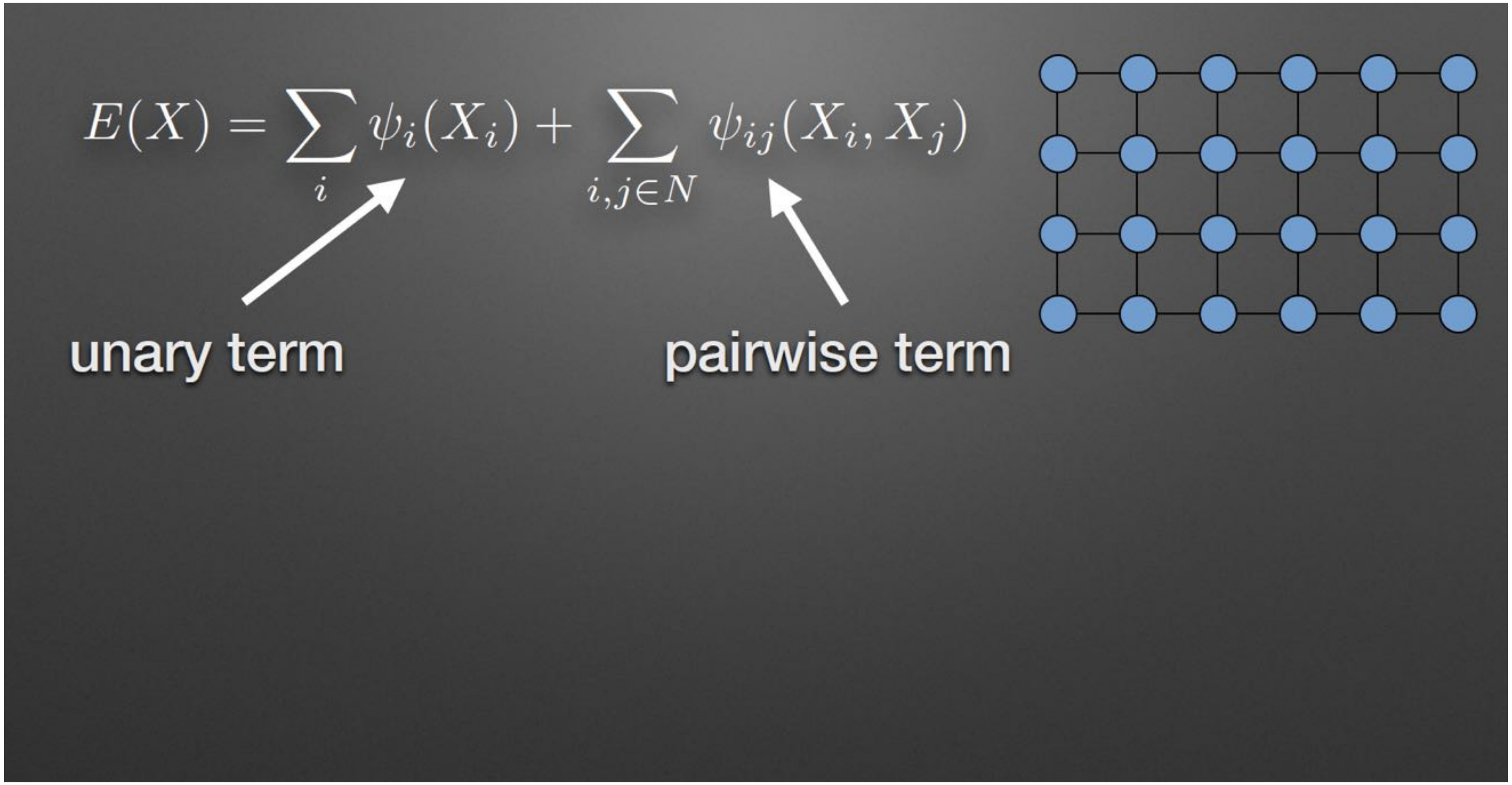
slide credit: Philipp Krähenbühl

Dense Random Fields



slide credit: Philipp Krähenbühl

Dense Random Fields

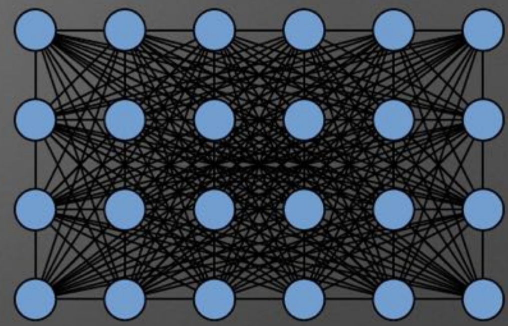


slide credit: Philipp Krähenbühl

Dense Random Fields

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i,j \in \cancel{N}} \psi_{ij}(X_i, X_j)$$

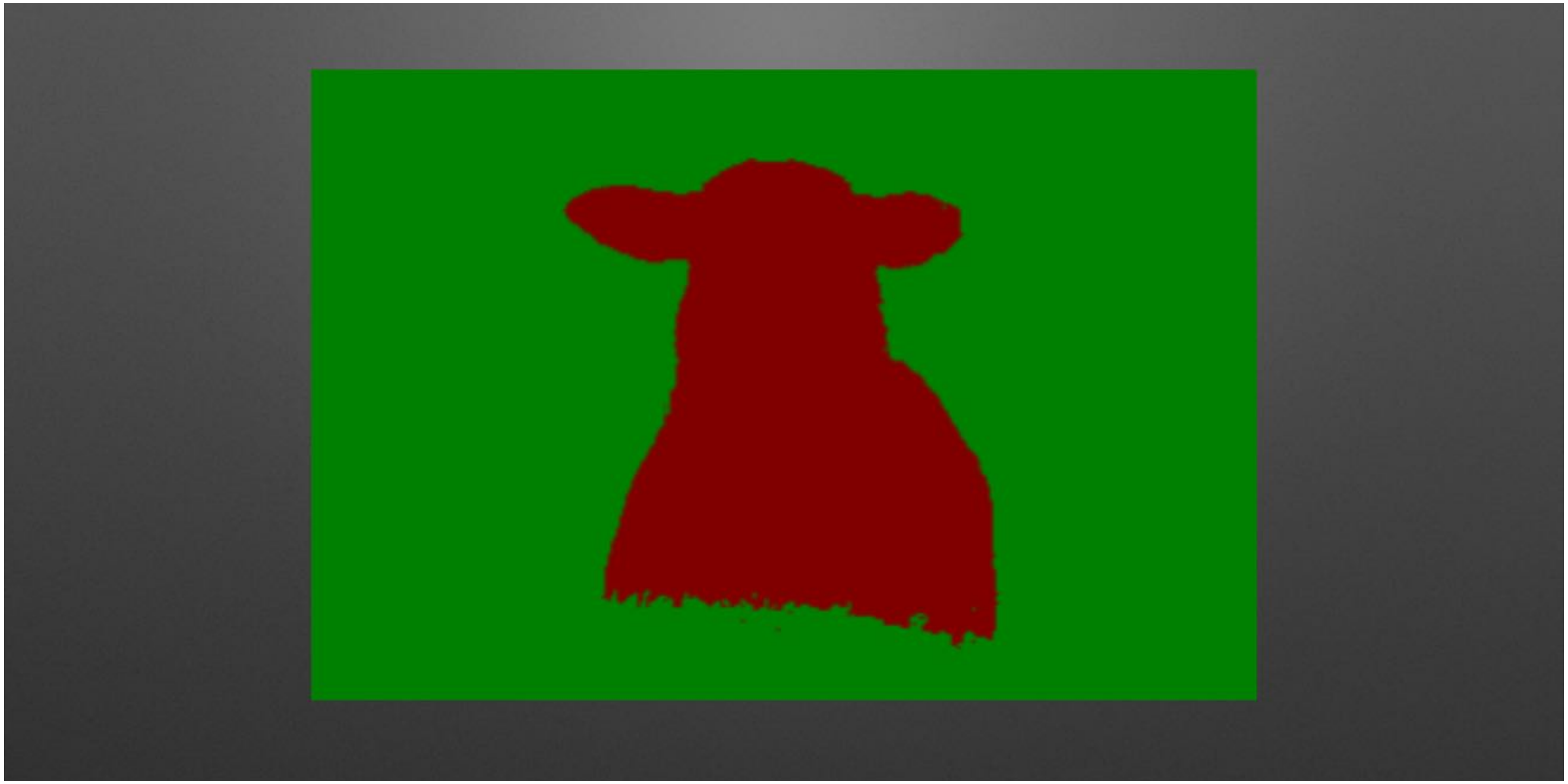
unary term ↗ pairwise term ↖



- Every node is connected to every other node
  - Connections weighted differently

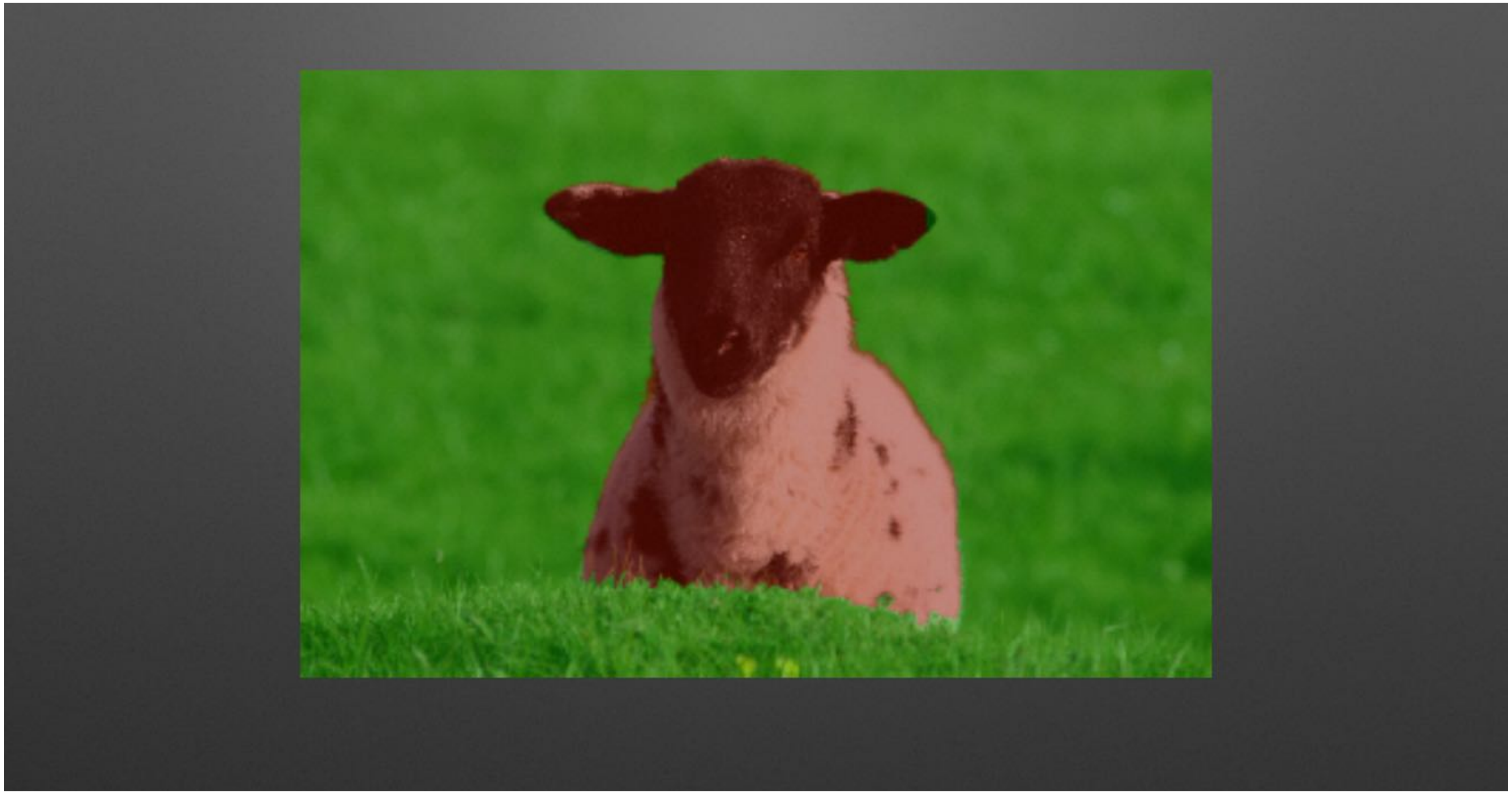
slide credit: Philipp Krähenbühl

Dense Random Fields



slide credit: Philipp Krähenbühl

Dense Random Fields



slide credit: Philipp Krähenbühl

Dense Random Fields

Pros:

- Long range interactions
- No shrinking bias

Dense Random Fields

Pros:

- Long range interactions
- No shrinking bias
- Probabilistic interpretation
- Parameter learning
- Combine with other models

Dense Random Fields

Cons:

- Very large model
  - 50’000 - 100’000 variables
  - billions pairwise terms
- Traditional inference very slow
  - MCMC “converges” in 36h
  - GraphCuts and alpha-exp.: no convergence in 3 days

Dense Random Fields

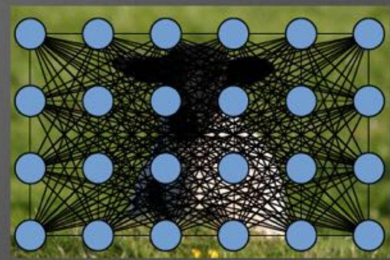
- Efficient inference
  - 0.2s / image
- Pairwise term
  - linear combination of Gaussians

Dense Random Fields

$$E(X) = \sum_{i} \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$

Dense Random Fields

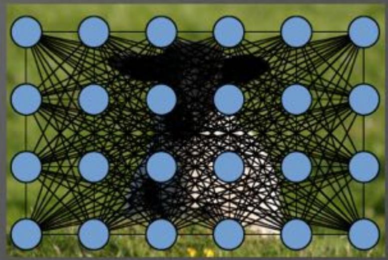
$$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$



$$\psi_{ij}(X_i, X_j) = \sum_m k^{(m)}(f_i, f_j)\, \mu^{(m)}(X_i, X_j)$$

Dense Random Fields

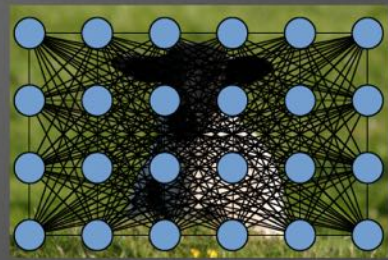
$$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$



$$\psi_{ij}(X_i, X_j) = \sum_m k^{(m)}(f_i, f_j)\, \mu^{(m)}(X_i, X_j)$$

Gaussian kernel $k^{(m)}$

Dense Random Fields



$$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$

$$\psi_{ij}(X_i, X_j) = \sum_m k^{(m)}(f_i, f_j)\, \mu^{(m)}(X_i, X_j)$$

Gaussian kernel $k^{(m)}$

$\otimes$

Label compatibility $\mu^{(m)}$

μ	GRASS	SHEEP	WATER	...
GRASS	0	1	1	...
SHEEP	1	0	10	...
WATER	1	10	0	...
...	...	...	...	0

Dense Random Fields

$$\psi_{ij}(X_i, X_j) = \mu_1(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\alpha^2} - \frac{|c_i - c_j|^2}{2\sigma_\beta^2}\right) +$$

$$\mu_2(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\gamma^2}\right)$$

Dense Random Fields

$$\psi_{ij}(X_i, X_j) = \mu_1(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\alpha^2} - \frac{|c_i - c_j|^2}{2\sigma_\beta^2}\right) + \mu_2(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\gamma^2}\right)$$

- Label compatibility
  - Potts model: μ(Xi,Xj) = [Xi≠Xj]

μ	GRASS	SHEEP	WATER	...
GRASS	0	1	1	1
SHEEP	1	0	1	1
WATER	1	1	0	1
...	1	1	1	0

Dense Random Fields

$$\psi_{ij}(X_i, X_j) = \mu_1(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\alpha^2} - \frac{|c_i - c_j|^2}{2\sigma_\beta^2}\right) +$$

$$\mu_2(X_i, X_j) \exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\gamma^2}\right)$$

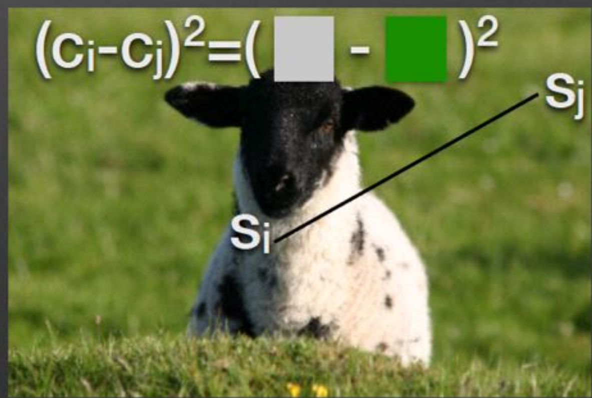
- Label compatibility
  - Potts model: μ(Xi,Xj) = [Xi≠Xj]
  - Learned from data

μ	GRASS	SHEEP	WATER	...
GRASS	0	?	?	?
SHEEP	?	0	?	?
WATER	?	?	0	?
...	?	?	?	0

Dense Random Fields

$$\psi_{ij}(X_i, X_j) = \mu_1(X_i, X_j)\exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\alpha^2} - \frac{|c_i - c_j|^2}{2\sigma_\beta^2}\right) + \mu_2(X_i, X_j)\exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\gamma^2}\right)$$

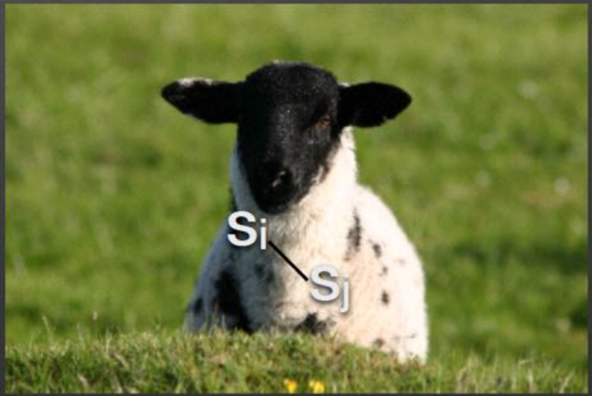
- Label compatibility
  - Potts model: μ(Xi,Xj) = [Xi≠Xj]
  - Learned from data
- Appearance kernel
  - Color—sensitive



Dense Random Fields

$$\psi_{ij}(X_i, X_j) = \mu_1(X_i, X_j)\exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\alpha^2} - \frac{|c_i - c_j|^2}{2\sigma_\beta^2}\right) + \mu_2(X_i, X_j)\exp\left(-\frac{|s_i - s_j|^2}{2\sigma_\gamma^2}\right)$$

- Label compatibility
  - Potts model: μ(Xi,Xj) = [Xi≠Xj]
  - Learned from data
- Appearance kernel
  - Color—sensitive
- Local smoothness
  - Discourages single pixel noise

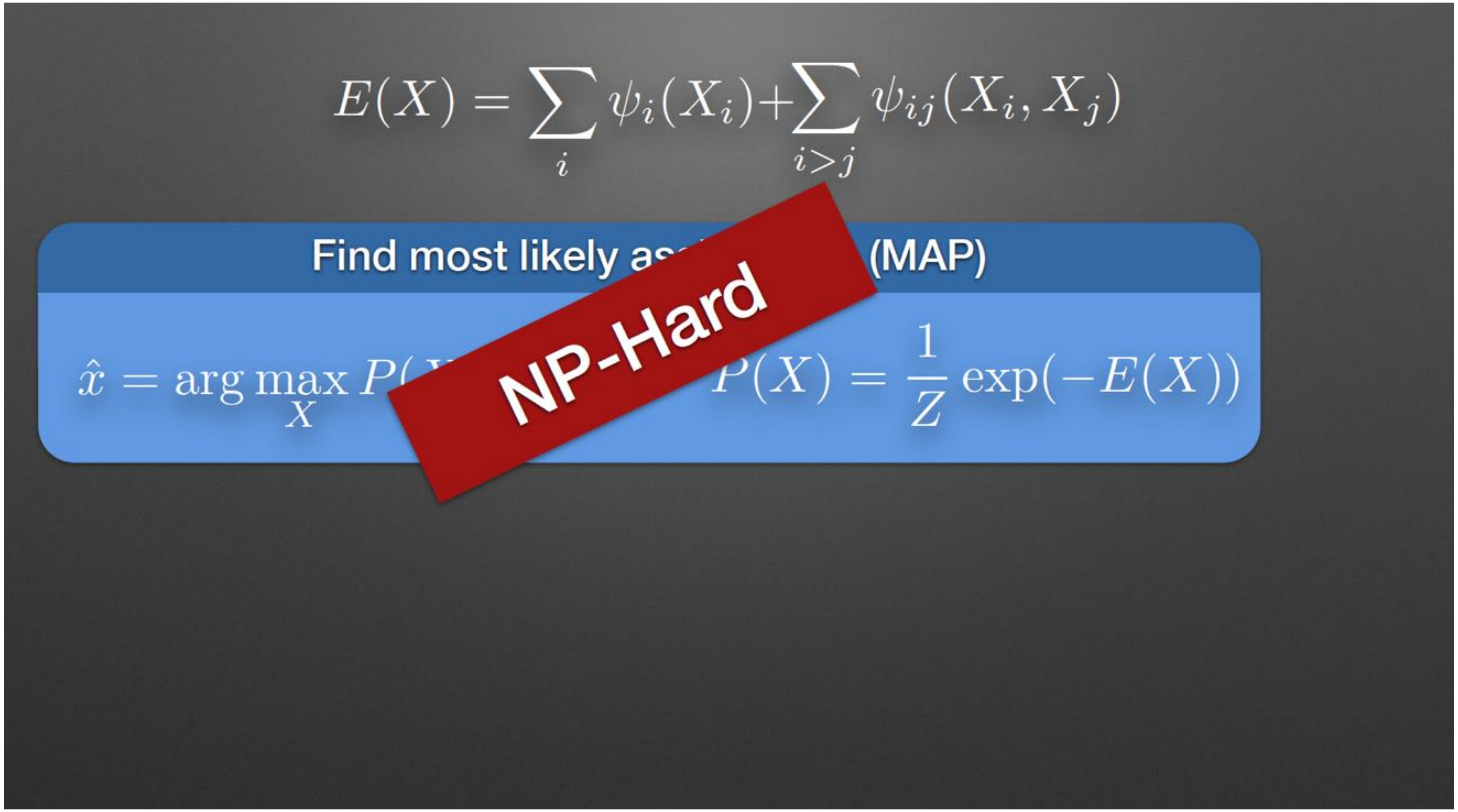


Efficient Inference

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$

Find most likely assignment (MAP)

$$\hat{x} = \arg\max_X P(X)$$ where $$P(X) = \frac{1}{Z}\exp(-E(X))$$


$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$
Find most likely
(MAP)
$\hat{x} = \arg\max_X P($
$P(X) = \frac{1}{Z} \exp(-E(X))$
NP-Hard

Efficient Inference

$$E(X) = \sum_i \psi_i(X_i) + \sum_{i>j} \psi_{ij}(X_i, X_j)$$

Find most likely as… (MAP)

$\hat{x} = \arg\max_X P($ …

NP-Hard

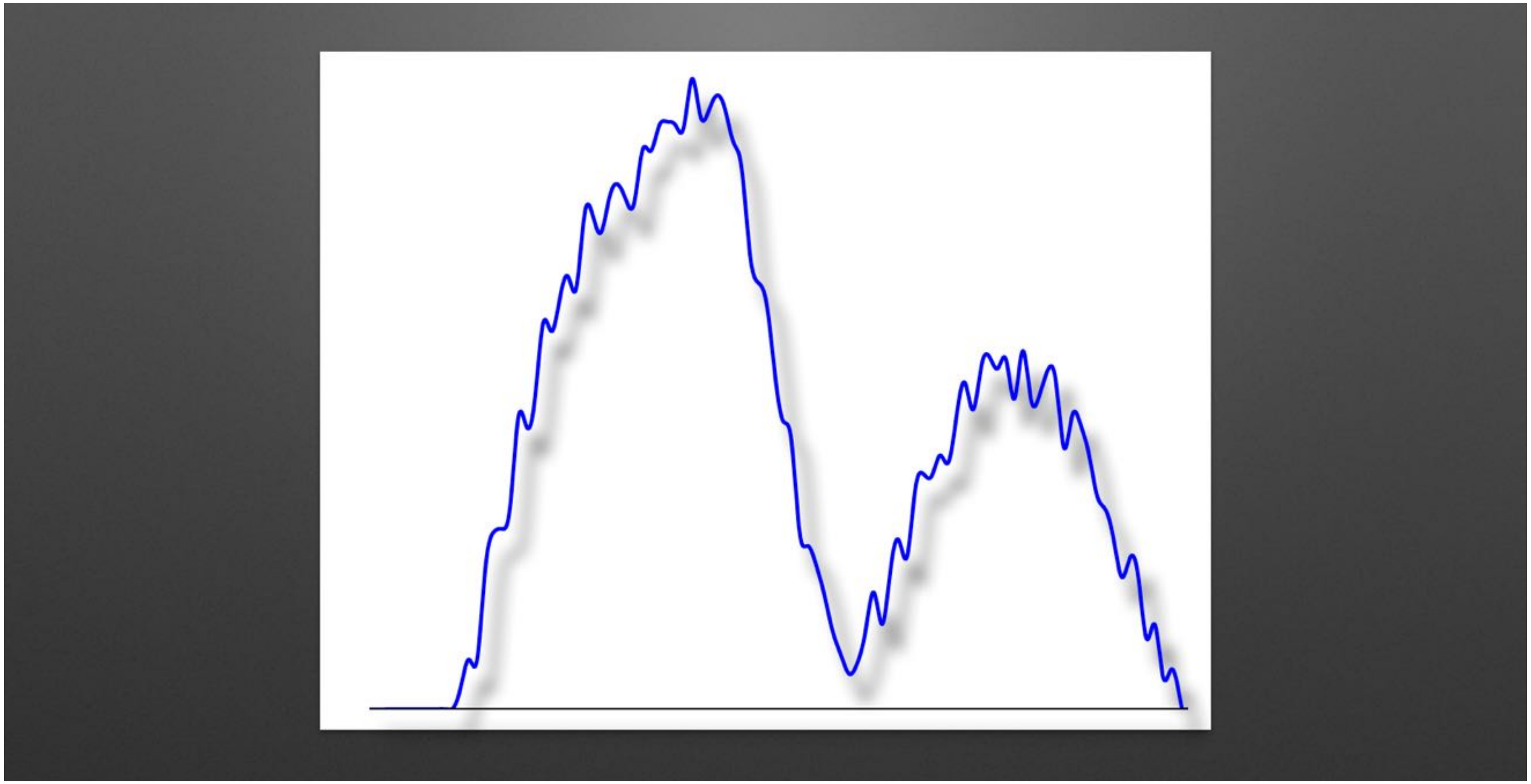
$$P(X) = \frac{1}{Z}\exp(-E(X))$$

Mean Field approximation

Find $Q(X)=\prod_i Q_i(X_i)$ close to P(X) in terms of KL-divergence D(Q||P)

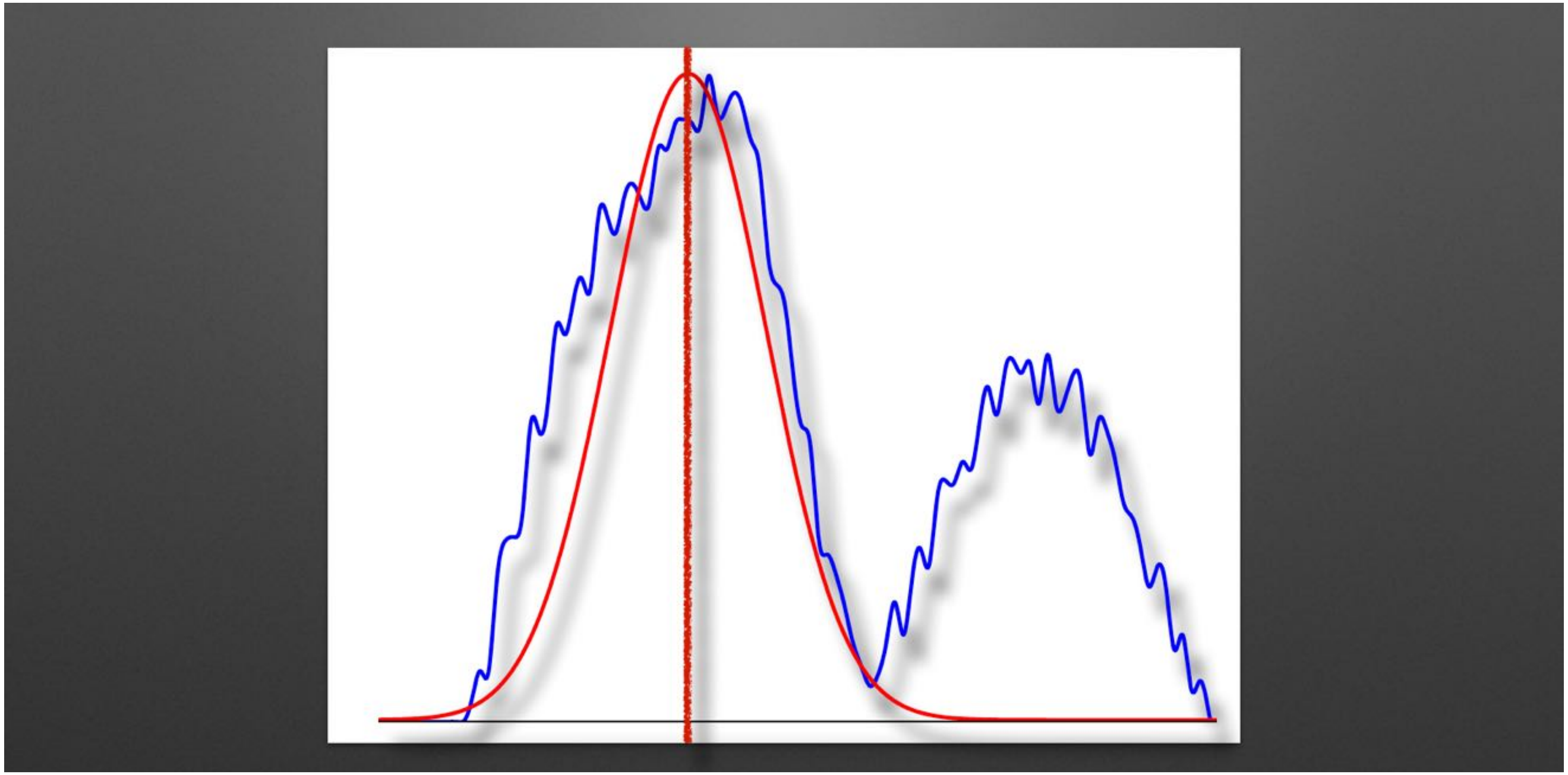
$$\hat{x} \approx \arg\max_X Q(X)$$

Mean-Field Approximation



slide credit: Philipp Krähenbühl

Mean-Field Approximation



slide credit: Philipp Krähenbühl

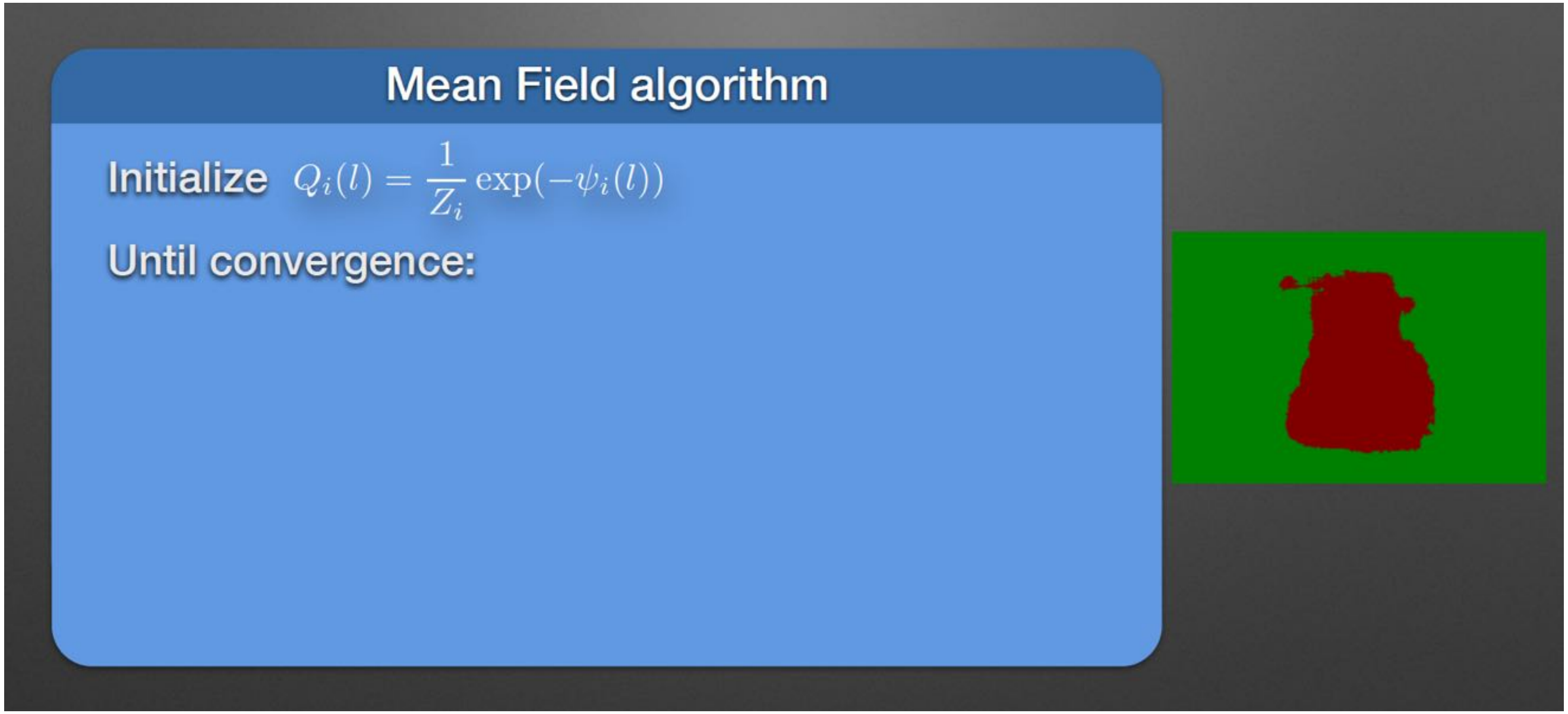
Efficient Inference

Mean Field algorithm

Initialize $Q_i(l) = \frac{1}{Z_i} \exp(-\psi_i(l))$

slide credit: Philipp Krähenbühl

Efficient Inference



slide credit: Philipp Krähenbühl

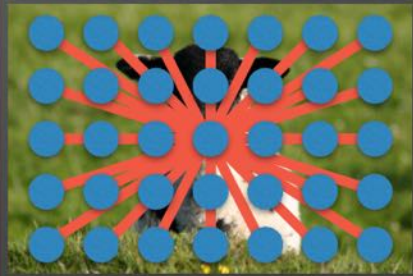
Efficient Inference

Mean Field algorithm

Initialize $Q_i(l) = \frac{1}{Z_i}\exp(-\psi_i(l))$

Until convergence:

- Message passing: $\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$



slide credit: Philipp Krähenbühl

Efficient Inference

Mean Field algorithm

Initialize $Q_i(l) = \frac{1}{Z_i} \exp(-\psi_i(l))$

Until convergence:

- Message passing: $\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$
- Compatibility transform: $\hat{Q}_i^{(m)}(l') = \sum_l \mu^{(m)}(l', l) \tilde{Q}_i^{(m)}(l)$

μ	GRASS	SHEEP	WATER	...
GRASS	0	1	1	...
SHEEP	1	0	10	...
WATER	1	10	0	...
...	...	...	...	0

slide credit: Philipp Krähenbühl

Efficient Inference

Mean Field algorithm

Initialize $Q_i(l) = \frac{1}{Z_i} \exp(-\psi_i(l))$

Until convergence:

- Message passing: $\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$
- Compatibility transform: $\hat{Q}_i^{(m)}(l') = \sum_l \mu^{(m)}(l', l) \tilde{Q}_i^{(m)}(l)$
- Local update: $Q_i(l) = \exp(-\psi_i(l) - \sum_m \hat{Q}_i^{(m)}(l))$

slide credit: Philipp Krähenbühl

Efficient Inference

Mean Field algorithm

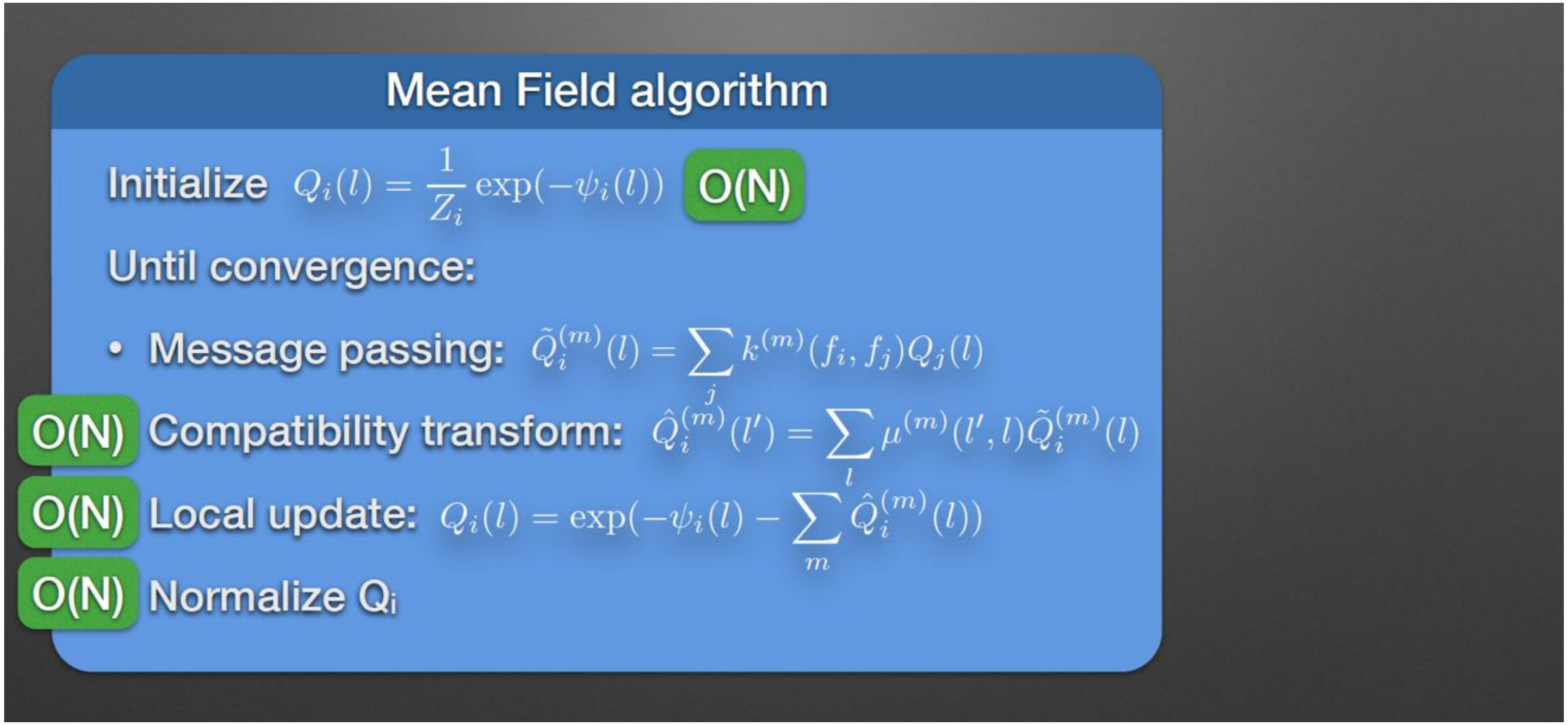
Initialize $Q_i(l) = \frac{1}{Z_i}\exp(-\psi_i(l))$

Until convergence:

- Message passing: $\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$
- Compatibility transform: $\hat{Q}_i^{(m)}(l') = \sum_l \mu^{(m)}(l', l)\tilde{Q}_i^{(m)}(l)$
- Local update: $Q_i(l) = \exp(-\psi_i(l) - \sum_m \hat{Q}_i^{(m)}(l))$
- Normalize Q_i

slide credit: Philipp Krähenbühl

Efficient Inference



slide credit: Philipp Krähenbühl

Efficient Inference

Mean Field algorithm

Initialize $Q_i(l) = \frac{1}{Z_i}\exp(-\psi_i(l))$ O(N)

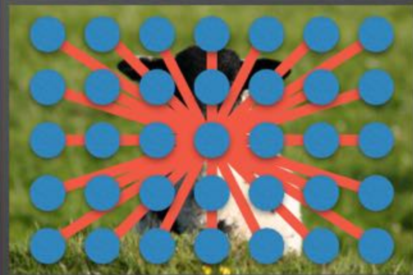
Until convergence:

O(N²) Message passing: $\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$

O(N) Compatibility transform: $\hat{Q}_i^{(m)}(l') = \sum_l \mu^{(m)}(l', l)\tilde{Q}_i^{(m)}(l)$

O(N) Local update: $Q_i(l) = \exp(-\psi_i(l) - \sum_m \hat{Q}_i^{(m)}(l))$

O(N) Normalize Q_i



slide credit: Philipp Krähenbühl

Efficient Message Passing

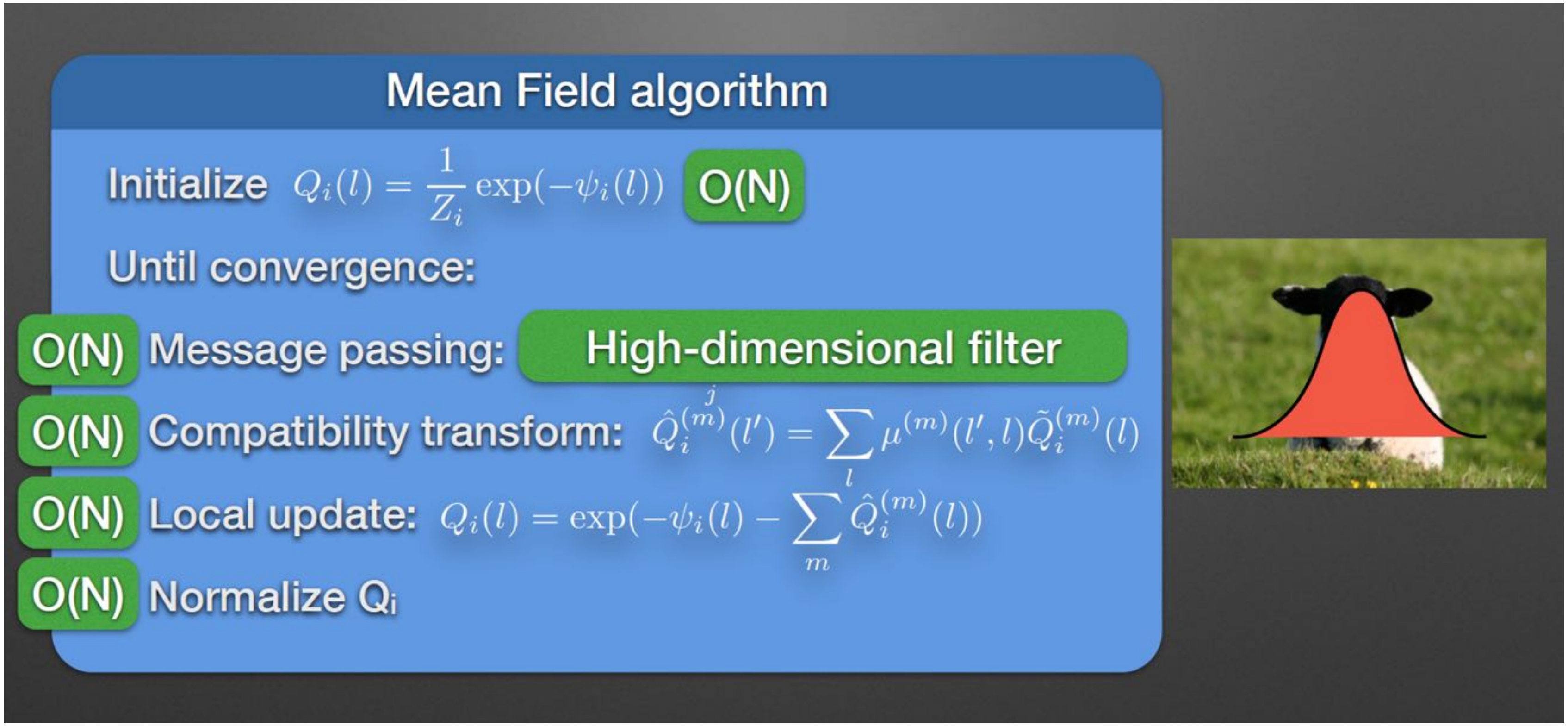
- Update all variables simultaneously

$$\tilde{Q}_i^{(m)}(l) = \sum_j k^{(m)}(f_i, f_j) Q_j(l)$$

- Gaussian Convolution
  - Efficient approximation

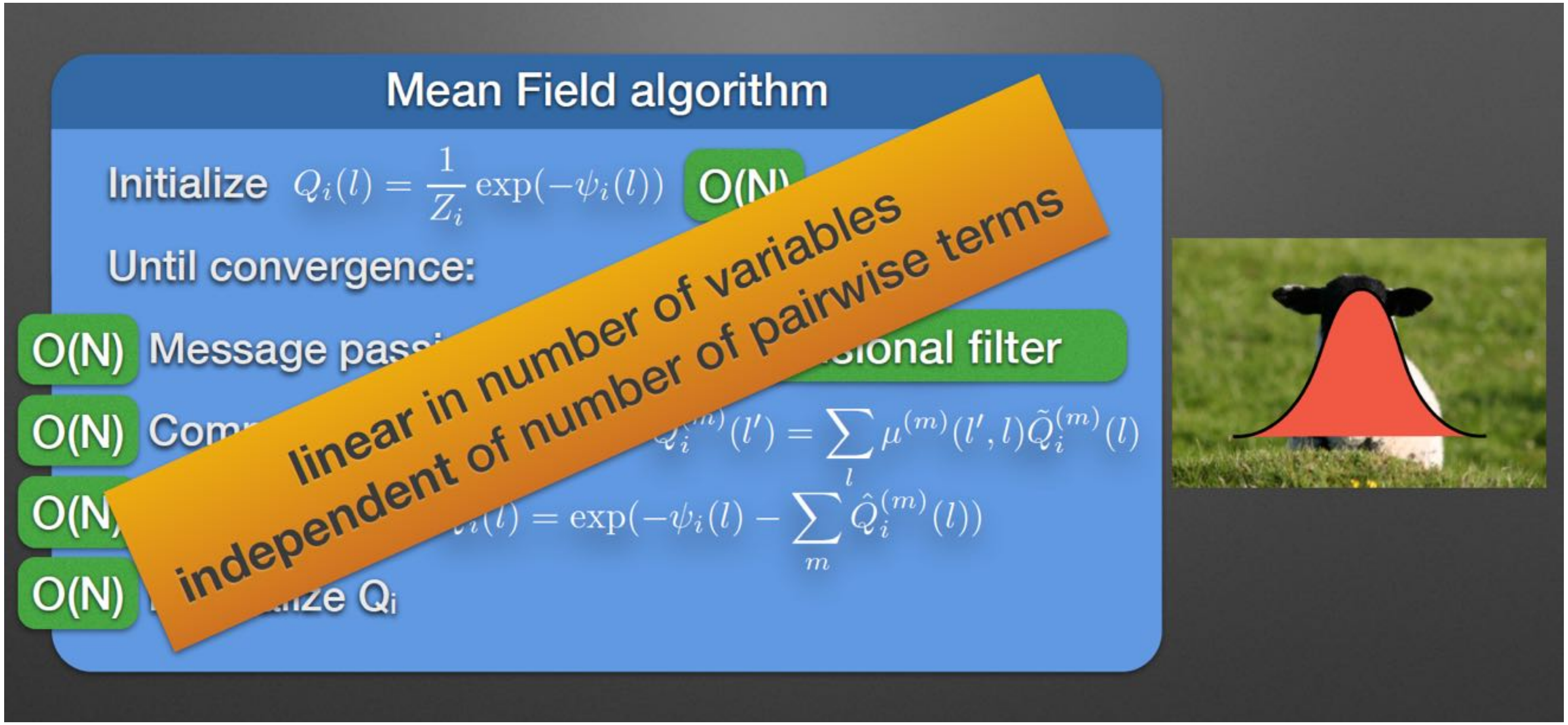
slide credit: Philipp Krähenbühl

Efficient Inference



slide credit: Philipp Krähenbühl

Efficient Inference



slide credit: Philipp Krähenbühl

Parallel Mean-Field

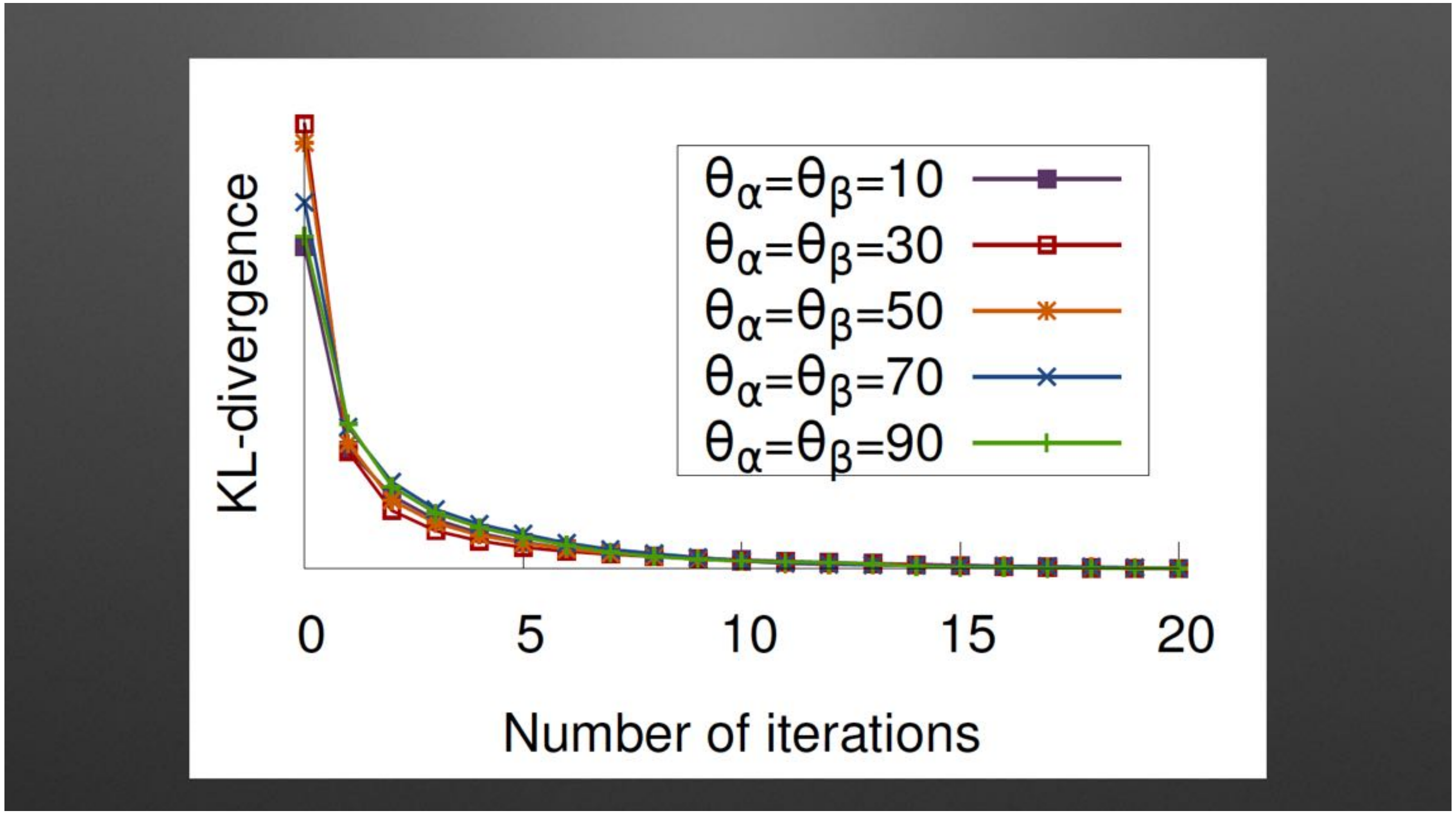
- Not guaranteed to converge for general models
- Guaranteed to converge for fully-connected models with negative definite label compatibility
  - Potts models
  - L1 norms
  - ...
- Proof see Thesis or [3]
  - Reduction of Parallel Mean-Field to CCCP

[3] Parameter Learning and Convergent Inference for Dense Random Fields, Krähenbühl and Koltun, ICML 2013

42

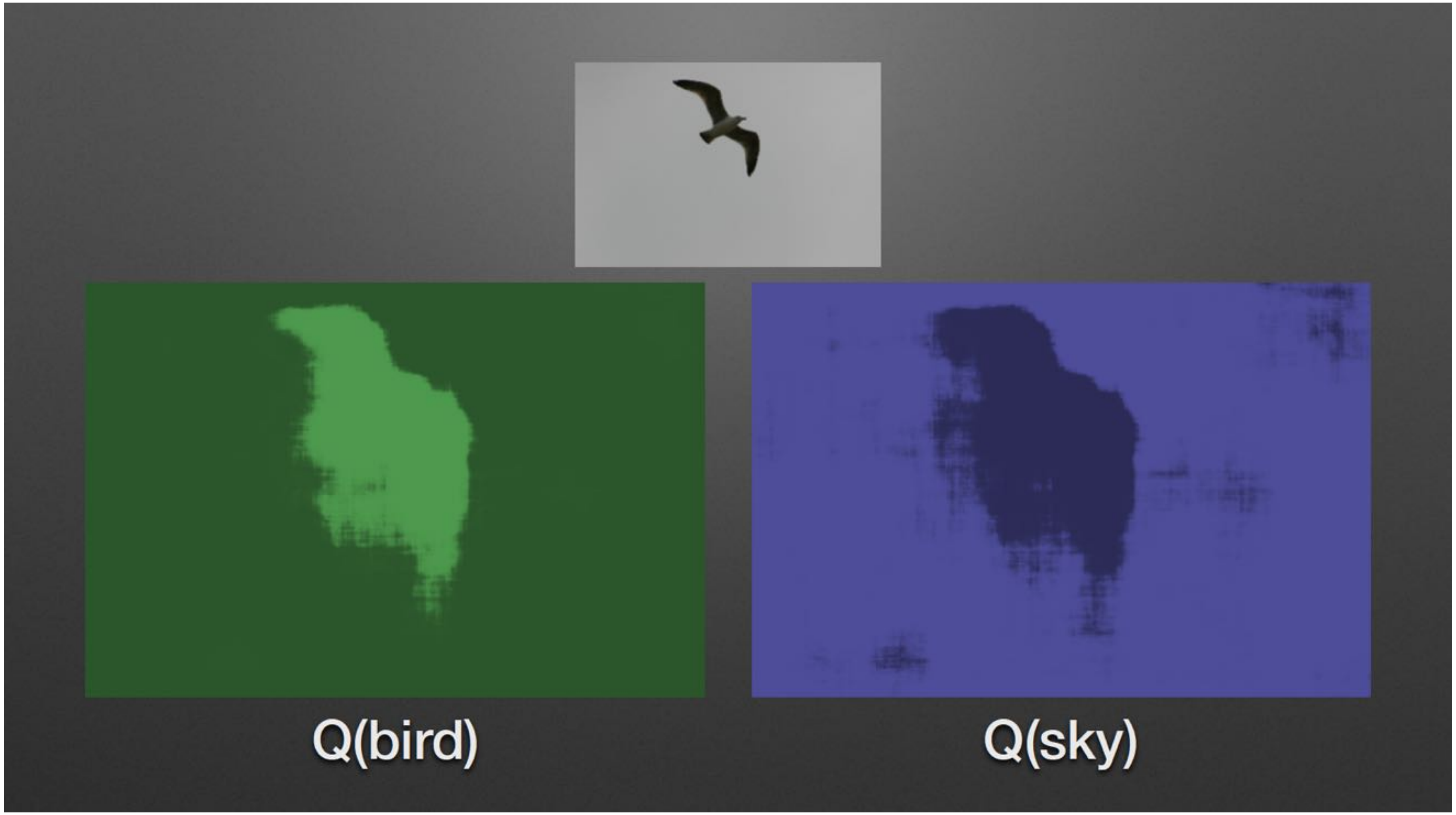
slide credit: Philipp Krähenbühl

How Fast Will it Converge

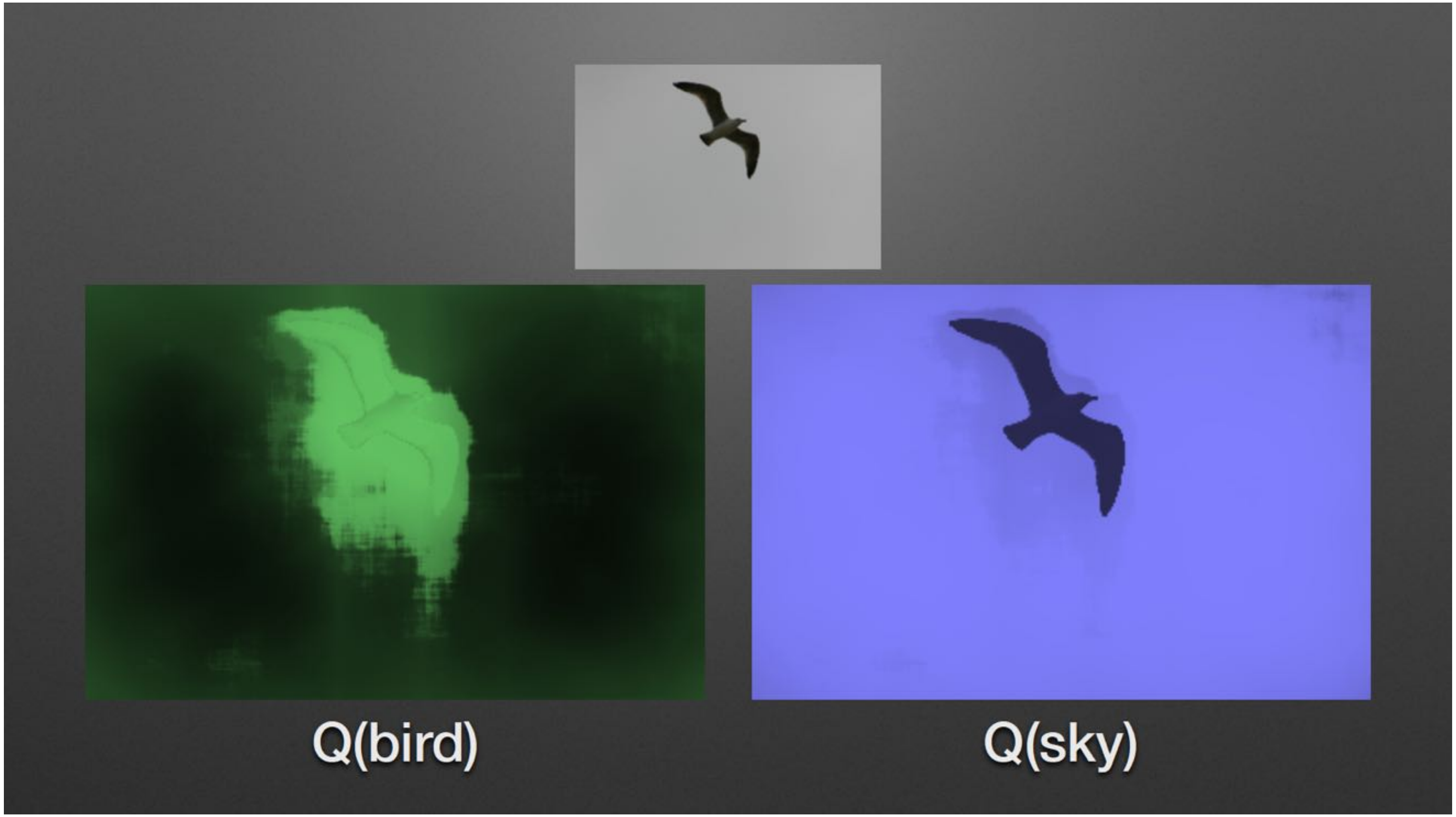


slide credit: Philipp Krähenbühl

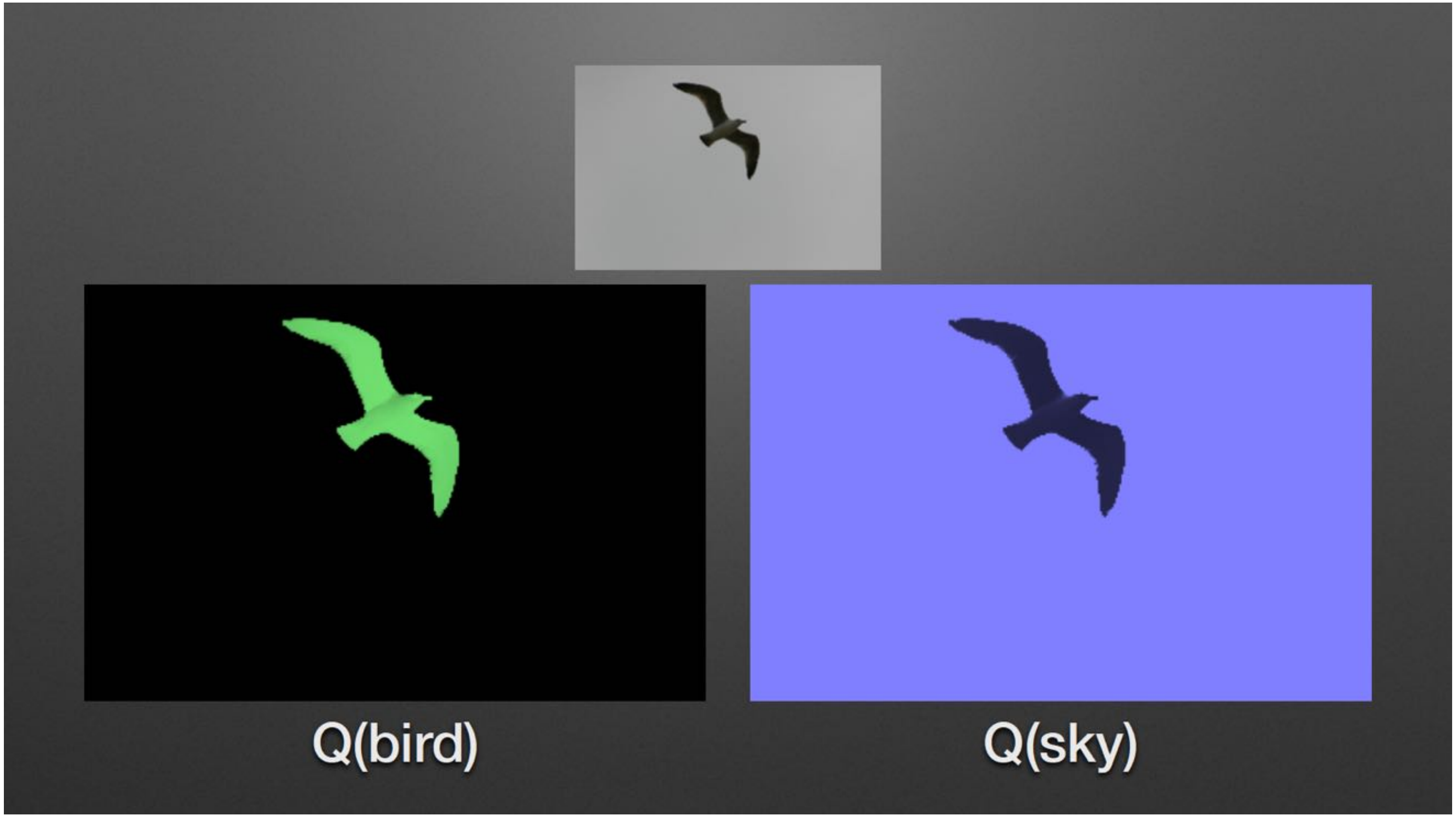
Iteration 0



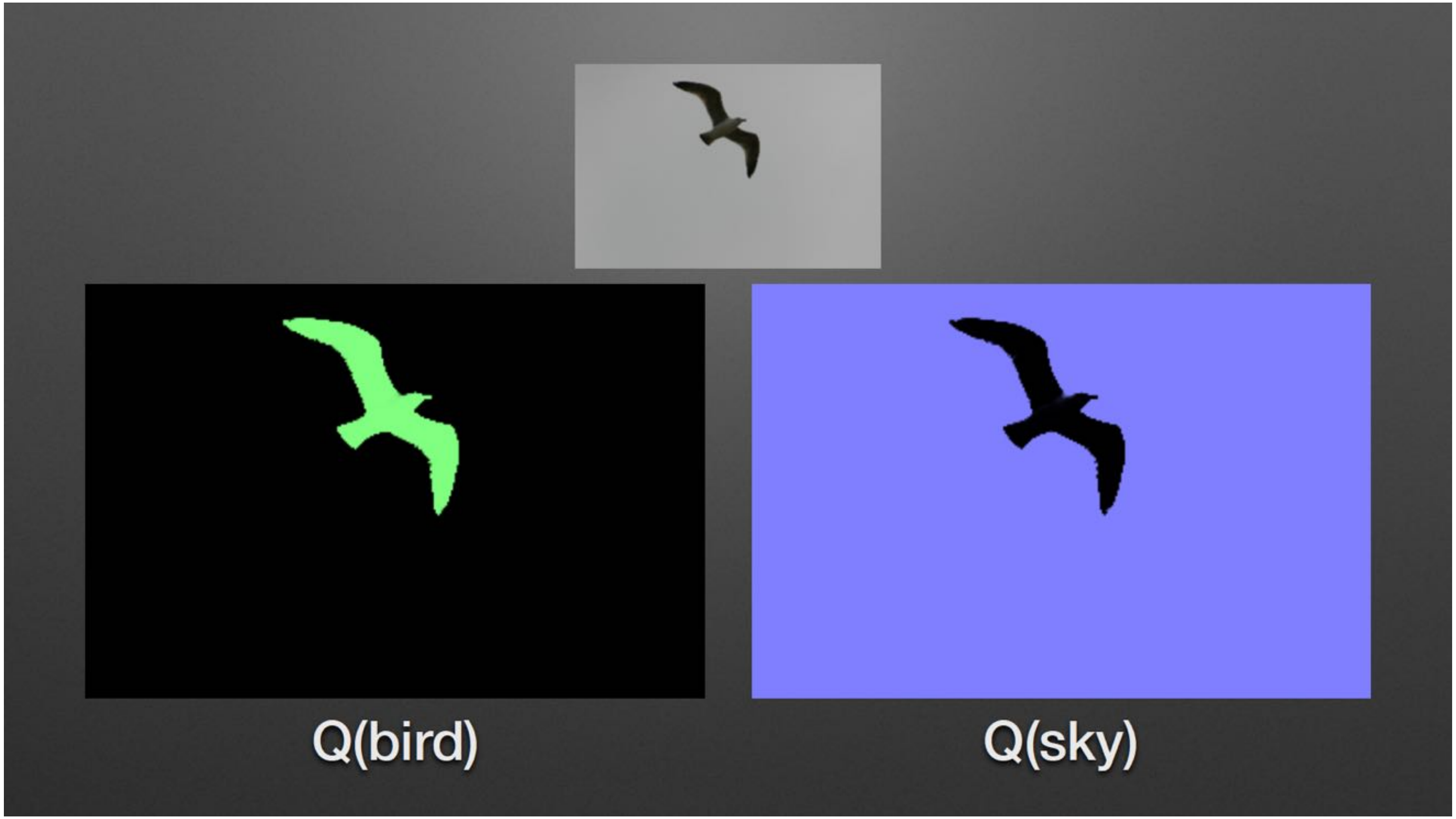
Iteration1



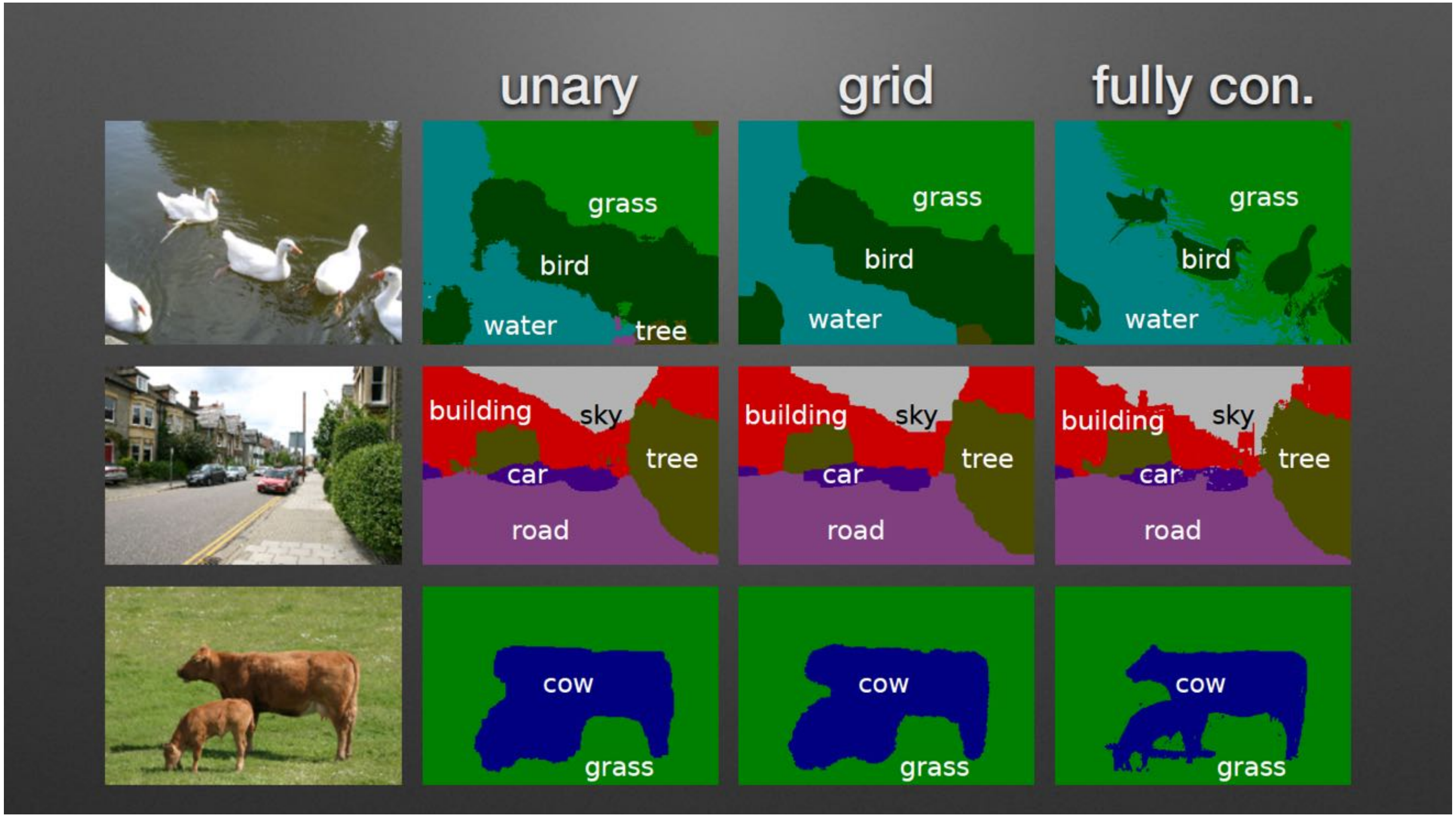
Iteration 2



Iteration 10



Results MSRC

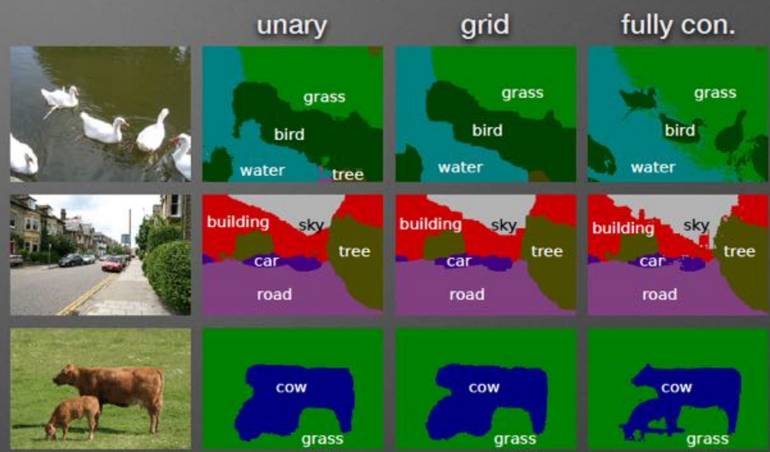


Results MSRC

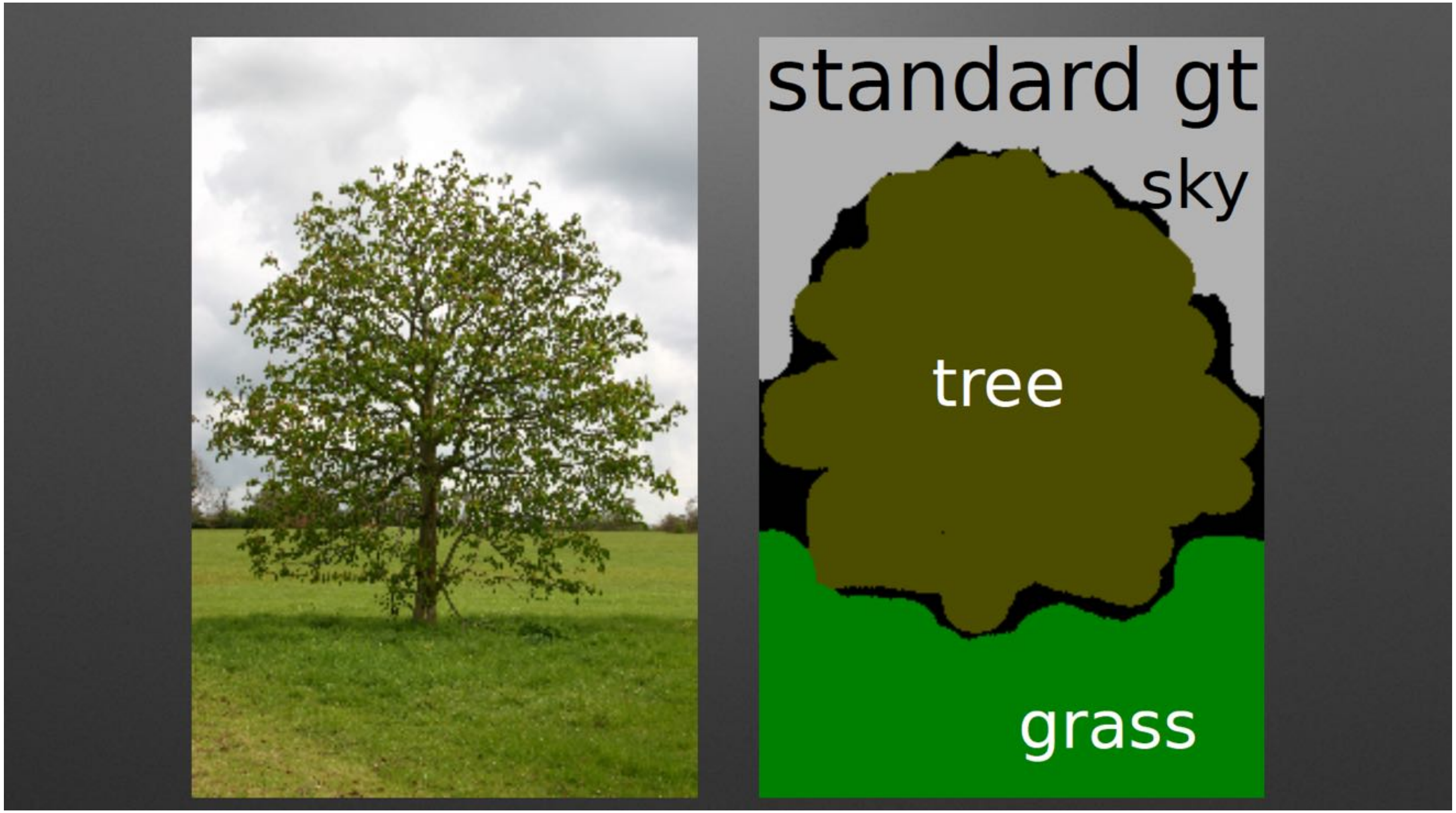
MSRC dataset

- 591 images
- 21 classes

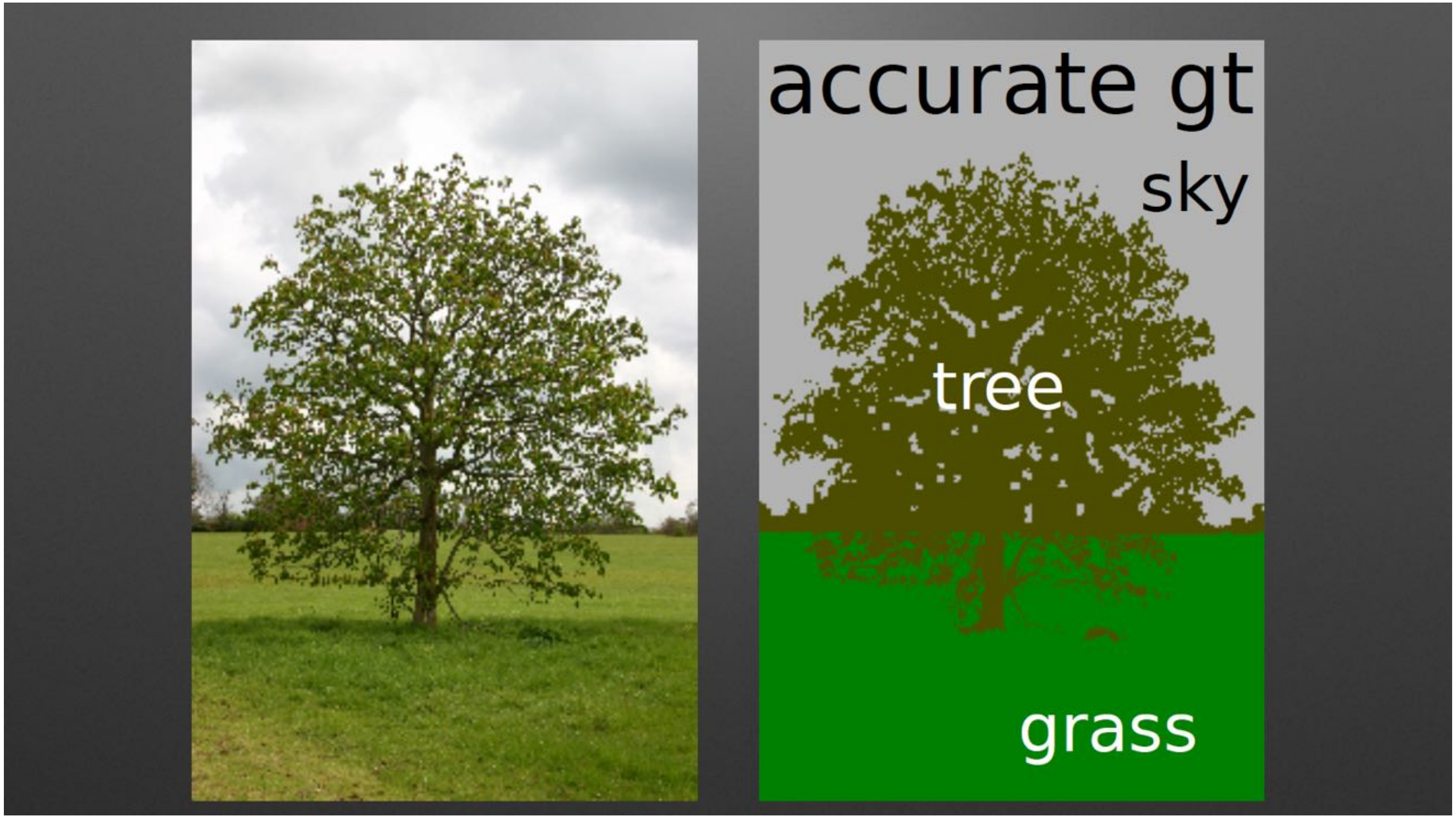
	TIME	GLOBAL	AVERAGE
UNARY	-	84.0	76.6
GRID CRF	1s	84.6	77.2
FC CRF	0.2s	**86.0**	**78.3**
FILTER	**0.05s**	85.0	77.5



Results MSRC



Results MSRC

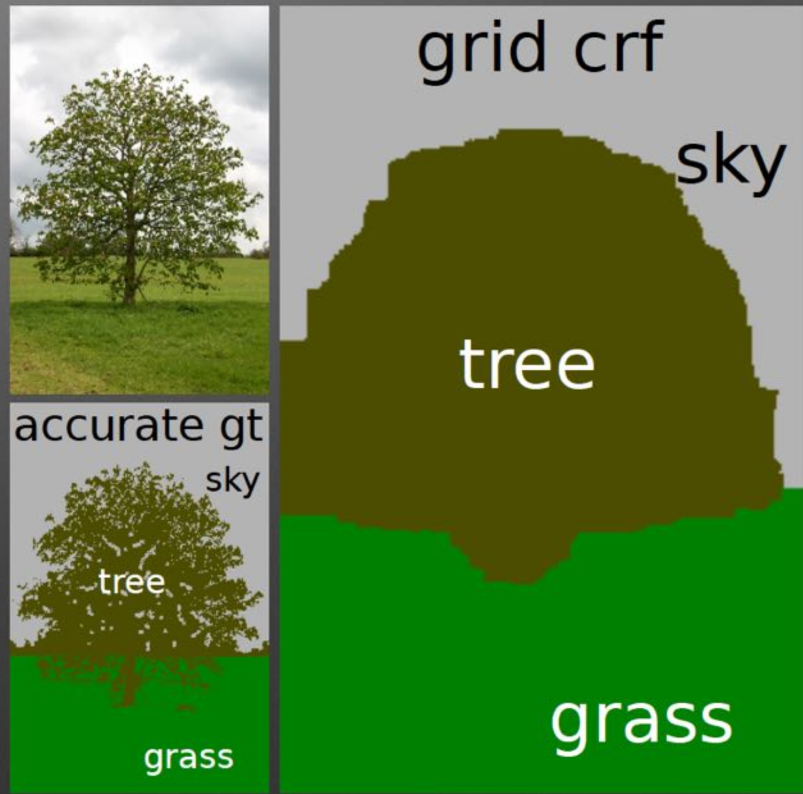


Results MSRC

MSRC Accurate annotations

- 94 images
- hand annotated (30 min each)
- unary train on standard anno.
- 5-fold cross validation

	GLOBAL	AVERAGE
UNARY	83.2±1.5	80.6±2.3
GRID CRF	84.8±1.5	82.4±1.8
FC CRF	**88.2±0.7**	**84.7±0.7**

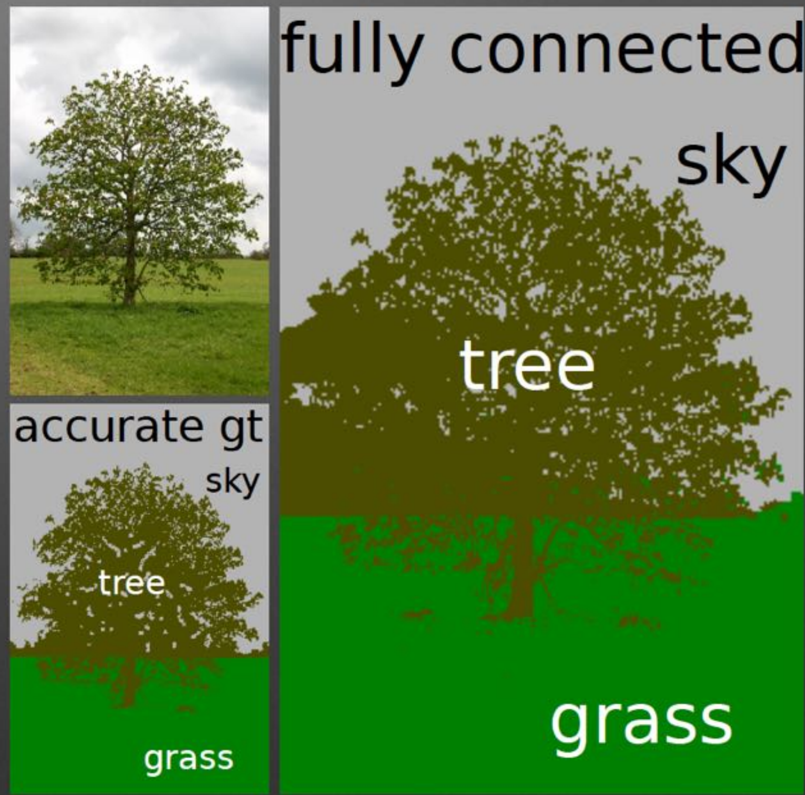


Results MSRC

MSRC Accurate annotations

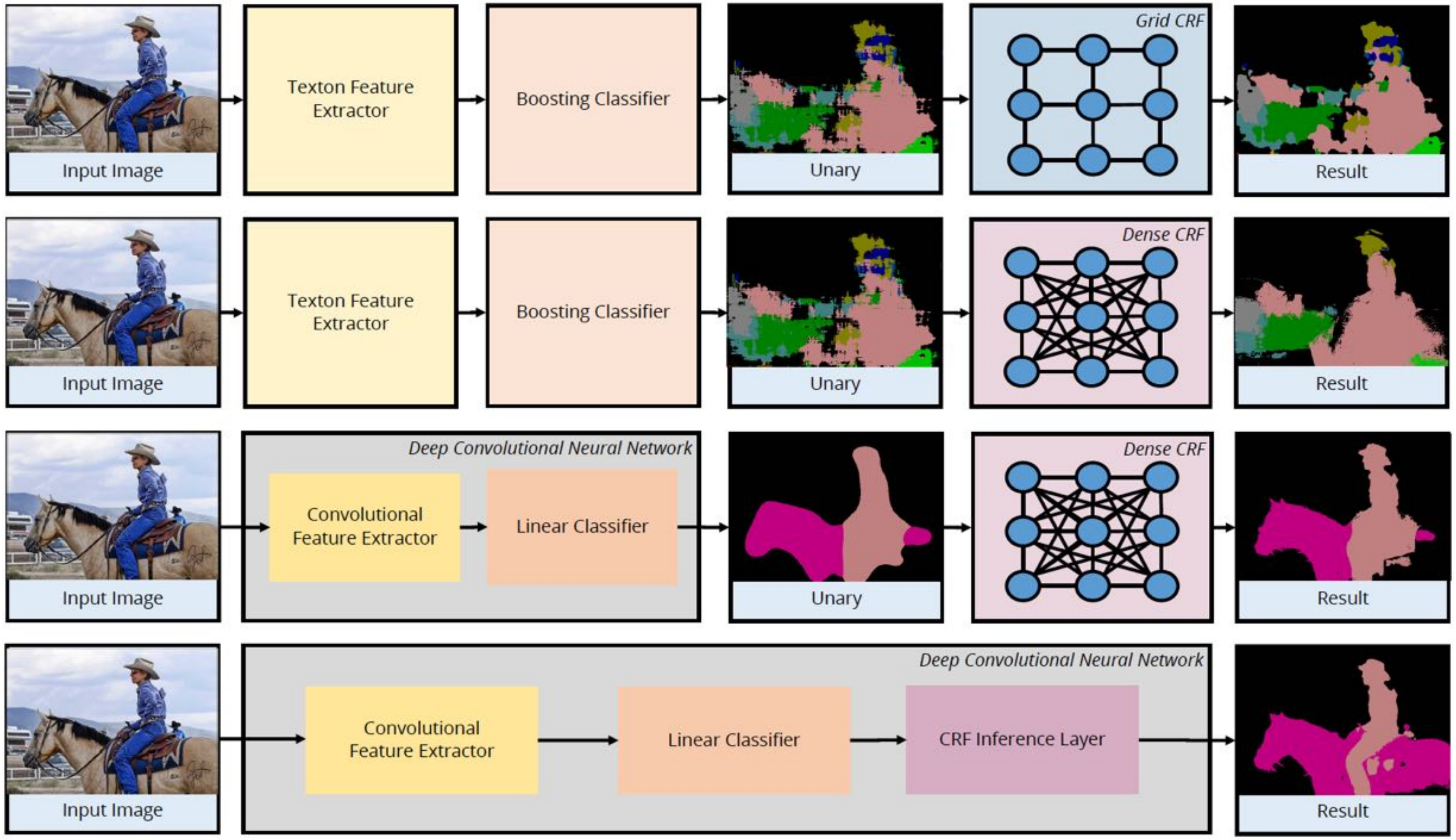
- 94 images
- hand annotated (30 min each)
- unary train on standard anno.
- 5-fold cross validation

	GLOBAL	AVERAGE
UNARY	83.2±1.5	80.6±2.3
GRID CRF	84.8±1.5	82.4±1.8
FC CRF	**88.2±0.7**	**84.7±0.7**



Pictorial Overview of Today's Lecture

image credit: paper [2]



Deep Convolutional Neural Networks...

- Top: (Sub-)Image Classification
- Bottom FCN (Fully Convolutional Neural Network)

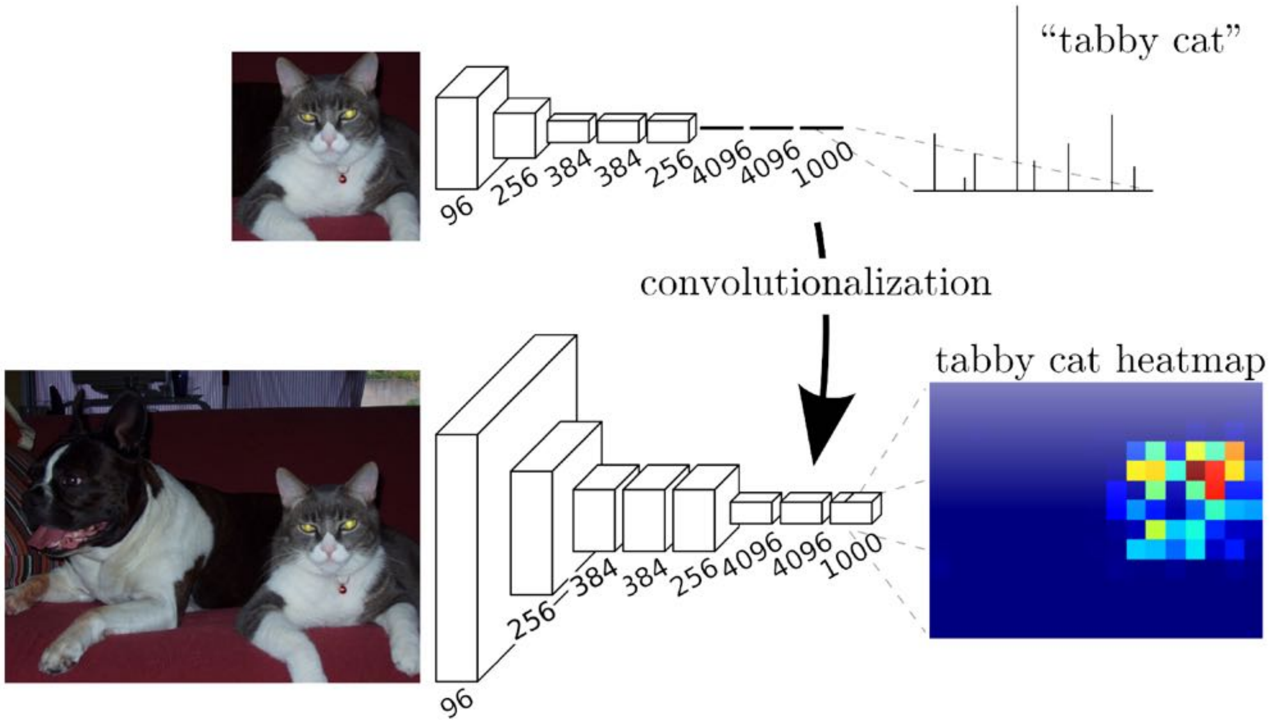


image credit: paper [2]

Pascal VOC Semantic Segmentation Results

- Block 1: no deep learning (DL)
- Block 2: using deep learning (DL)
  - but not CRF
- Block 3: using DL + CRF
  - but deep learning and CRF not trained jointly
- Questions:
  - how to benefit better from both?
  - how to jointly learn?
  - can we perform "end-to-end" training?

Method	IoU [%]	Base Network
Methods not using deep learning		
O2P [36]	47.8	–
Methods not using a CRF		
SDS [37]	51.6	AlexNet
FCN [6]	67.2	VGG
Zoom-out [38]	69.6	VGG
Methods using CRF for post-processing		
DeepLab [5]	71.6	VGG
EdgeNet [39]	73.6	VGG
BoxSup [40]	75.2	VGG
Dilated Conv [27]	75.3	VGG
Centrale Boundaries [41]	75.7	VGG
DeepLab Attention [42]	76.3	VGG
LRR [30]	79.3	ResNet
DeepLab v2 [43]	79.7	ResNet

table credit: paper [2]

Dense CRF - Mean Field Inference Algorithm

Algorithm 1 Mean field inference for Dense CRF [4], composed from common CNN operations.

$Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp(U_u(l'))} \exp(U_u(l))$ ▷ Initialization

while not converged **do**

$\tilde{Q}_u^{(m)}(l) \leftarrow \sum_{v \neq u} k^{(m)}(\mathbf{f}_u, \mathbf{f}_v) Q_v(l)$ for all m

▷ Message Passing

$\check{Q}_u(l) \leftarrow \sum_m w^{(m)} \tilde{Q}_u^{(m)}(l)$

▷ Weighting Filter Outputs

$\hat{Q}_u(l) \leftarrow \sum_{l' \in L} \mu(l, l') \check{Q}_u(l')$

▷ Compatibility Transform

$\breve{Q}_u(l) \leftarrow U_u(l) - \hat{Q}_u(l)$

▷ Adding Unary Potentials

$Q_u(l) \leftarrow \frac{1}{\sum_{l'} \exp\left(\breve{Q}_u(l')\right)} \exp\left(\breve{Q}_u(l)\right)$

▷ Normalizing

end while

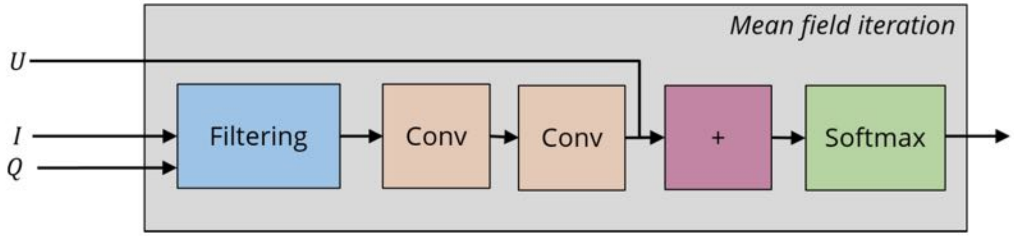


image credit: paper [2]

How Many Mean Field Iterations?

- Classically:
  - Iterate until convergence
- Here:
  - Fix the number of iterations (in the figure T) and simply concatenate
  - called "CRF-as-RNN"

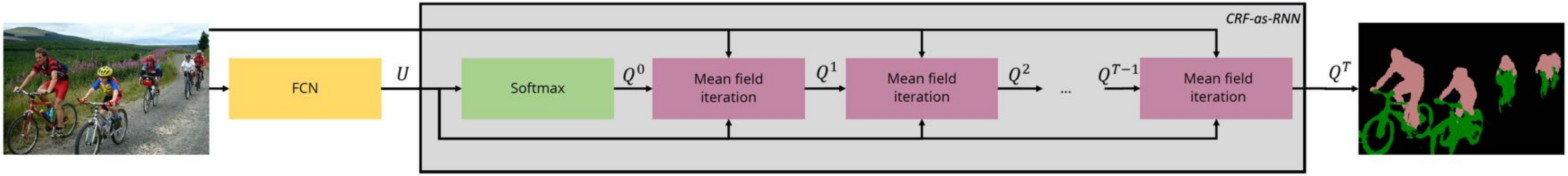


image credit: paper [2]

Pascal VOC Semantic Segmentation Results

- Block 1: no deep learning (DL)
- Block 2: using deep learning (DL)
  - but not CRF
- Block 3: using DL + CRF
  - but deep learning and CRF not trained jointly
- Block 4: end-to-end training of DL & CRF

Method	IoU [%]	Base Network
Methods not using deep learning		
O2P [36]	47.8	–
Methods not using a CRF		
SDS [37]	51.6	AlexNet
FCN [6]	67.2	VGG
Zoom-out [38]	69.6	VGG
Methods using CRF for post-processing		
DeepLab [5]	71.6	VGG
EdgeNet [39]	73.6	VGG
BoxSup [40]	75.2	VGG
Dilated Conv [27]	75.3	VGG
Centrale Boundaries [41]	75.7	VGG
DeepLab Attention [42]	76.3	VGG
LRR [30]	79.3	ResNet
DeepLab v2 [43]	79.7	ResNet
Methods with end-to-end CRFs		
CRF as RNN [7]	74.7	VGG
Deep Gaussian CRF [8]	75.5	VGG
Deep Parsing Network [44]	77.5	VGG
Context [32]	77.8	VGG
Higher Order CRF [33]	77.9	VGG
Deep Gaussian CRF [8]	80.2	ResNet

Pictorial Overview of Today's Lecture

image credit: paper [2]