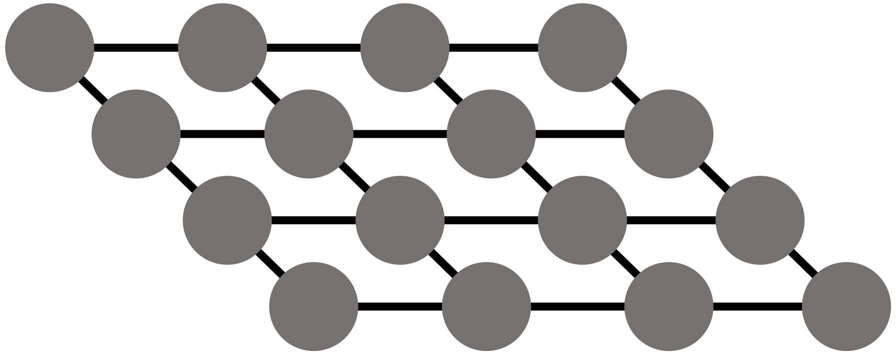
Graphical models for refinement

Last time: structured prediction

- Output space is *large* and *structured*
  - Large: cannot be enumerated
  - Structured: some outputs are *a priori* more likely than others
- Basic idea:
  - Define probabilistic model P(y|x)
    - Use domain knowledge of what prior should be
  - Estimate the most likely label at test time
    - $y^* = \arg\max P(y|x)$

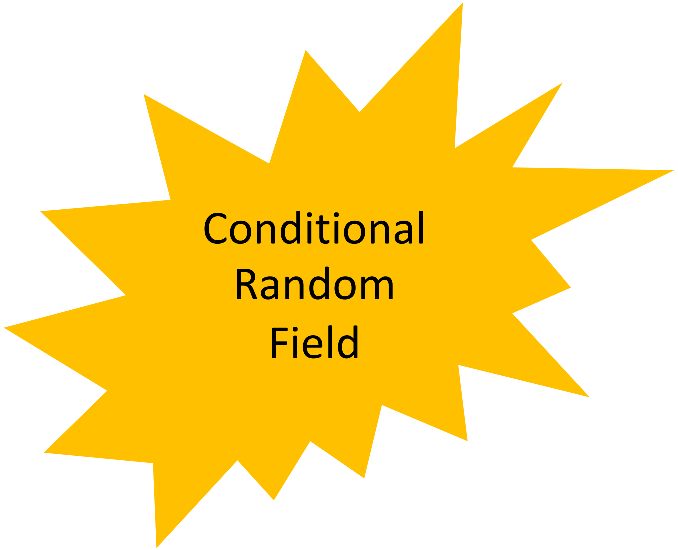
Defining probabilistic model for segmentation



$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} e^{-E(\mathbf{y},\mathbf{x})}$$

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x})$$

$$= \arg\min_{\mathbf{y}} E(\mathbf{y},\mathbf{x})$$



$$E(\mathbf{y},\mathbf{x}) = \sum_i \phi_i^{data}(y_i, \mathbf{x}) + \sum_{i,j \in \mathcal{N}} \phi_{ij}^{smooth}(y_i, y_j, \mathbf{x})$$

Example data terms

$$E(\mathbf{y}, \mathbf{x}) = \sum_i \phi_i^{data}(y_i, \mathbf{x}) + \sum_{i,j \in \mathcal{N}} \phi_{ij}^{smooth}(y_i, y_j, \mathbf{x})$$

- What should ϕ_i^{data} be?
- Also called *unary terms*
- Given an image $\boldsymbol{x}$ and a candidate label y_i for pixel i, score it
- Can be any classifier operating on individual pixels
- Can also be class scores output by a semantic segmentation network

Example smoothness terms

$$E(\mathbf{y},\mathbf{x}) = \sum_i \phi_i^{data}(y_i,\mathbf{x}) + \sum_{i,j\in\mathcal{N}} \phi_{ij}^{smooth}(y_i,y_j,\mathbf{x})$$

- What should ϕ_{ij}^{smooth} be?
- Also called *binary terms*
- Given an image $\boldsymbol{x}$ and a candidate label y_i for pixel i and candidate label y_j for pixel j, score it
  - Example 1: 0 if $y_i = y_j$, 1 otherwise
  - Example 2: 0 if $y_i = y_j$, $w_{ij}(\boldsymbol{x})$ otherwise, where $w_{ij}(\boldsymbol{x})$ captures color similarity between i and j
  - Example 3: can be the output of a neural network.

Example graph structure

$$E(\mathbf{y}, \mathbf{x}) = \sum_i \phi_i^{data}(y_i, \mathbf{x}) + \sum_{i,j \in \mathcal{N}} \phi_{ij}^{smooth}(y_i, y_j, \mathbf{x})$$

- What should $\mathcal{N}$ be?
- Which pixels should we connect?
- Neighboring pixels?
  - Will only ensure local smoothness: cannot handle e.g., occlusion
- All pixels?
- Pixels within a neighborhood?

MAP estimation

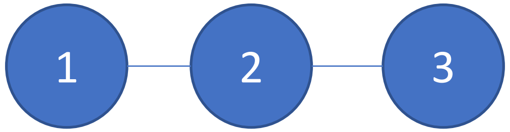
$$E(\mathbf{y},\mathbf{x}) = \sum_i \phi_i^{data}(y_i,\mathbf{x}) + \sum_{i,j\in\mathcal{N}} \phi_{ij}^{smooth}(y_i,y_j,\mathbf{x})$$

$$\boxed{\begin{aligned} P(\mathbf{y}|\mathbf{x}) &= \frac{1}{Z}e^{-E(\mathbf{y},\mathbf{x})} \\ &= \prod_i \psi_i^{data}(y_i,\mathbf{x}) \prod_{i,j\in\mathcal{N}} \psi_{ij}^{smooth}(y_i,y_j,\mathbf{x}) \end{aligned}}$$

$$\boxed{\begin{aligned} \mathbf{y}^* &= \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) \\ &= \arg\min_{\mathbf{y}} E(\mathbf{y},\mathbf{x}) \end{aligned}}$$

- When is this arg min solvable? When is it not?
- If not solvable, can we make approximations?

When is MAP estimation tractable?

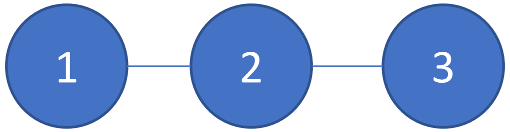


- Consider a tree
- MAP problem:

$$\arg\max_{y_1,y_2,y_3,y_4} \psi_1(y_1)\psi_2(y_2)\psi_3(y_3)\psi_{12}(y_1,y_2)\psi_{23}(y_2,y_3)$$

- Let's see how we might get the value of y_1^*

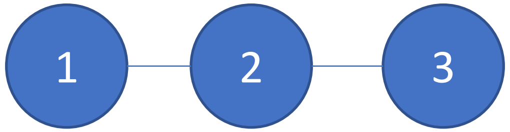
When is MAP estimation tractable?



- Consider a tree
- Rearranging:

$$\max_{y_1,y_2,y_3,y_4} \psi_1(y_1)\psi_{12}(y_1,y_2)\psi_2(y_2)\psi_{23}(y_2,y_3)\psi_3(y_3)$$

When is MAP estimation tractable?



- Consider a tree
- Push max in:

$$\max_{y_1} \psi_1(y_1) \left(\max_{y_2} \psi_{12}(y_1, y_2)\psi_2(y_2) \left(\max_{y_3} \psi_{23}(y_2, y_3)\psi_3(y_3) \right)\right)$$

$$= \max_{y_1} \psi_1(y_1) \left(\max_{y_2} \psi_{12}(y_1, y_2)\psi_2(y_2) m_{2\leftarrow 3}(y_2) \right)$$

$$= \max_{y_1} \psi_1(y_1) m_{1\leftarrow 2}(y_1)$$

Get y_1* from here!

Message passing / Belief propagation

- $m_{2\leftarrow 3}(y_2)$ and $m_{1\leftarrow 2}(y_1)$ are **messages** from 3 to 2 and 2 to 1 respectively
- $m_{i\leftarrow j}(y_i)$ captures what node j and other previous nodes think the label of i should be
- Final label for i: integrate message with unary potential on i
- How to get labels for other nodes?
  - Repeat, with memoization

Message passing/belief propagation

- General algorithm: belief propagation
- Each node starts with a **belief** $b_i(\cdot)$ about what its label should be
- Nodes send **messages** $m_{i \leftarrow j}(\cdot)$ to other nodes
- Beliefs get updated based on messages

Messsage passing / belief propagation

- Initialization

$$b_i(y_i) = \psi_i(y_i)$$
$$m_{i \leftarrow j}(y_i) = 1$$

- Repeat:

$$b_i(y_i) = \psi_i(y_i) \prod_{j \in \mathcal{N}(i)} m_{i \leftarrow j}(y_i)$$
$$m_{i \leftarrow j}(y_i) = \max_{y_j} \psi_j(y_j) \prod_{k \in \mathcal{N}(j), k \neq i} m_{k \leftarrow j}(y_j)$$

When is MAP estimation tractable?

- When graph is a tree
  - Designate one node as root
  - Send messages from leaves inward
  - Send messages from root outward
  - Done!
- When graph is not tree, belief propagation not guaranteed to converge and not guaranteed to be correct

Images

- Usually grid structure: not tractable
- Sometimes fully connected graphs
- Smoothness potential depends on labels and color and position difference between pixels

$$\phi_{ij}^{smooth}(y_i, y_j, \mathbf{x}) = \underbrace{\mu(y_i, y_j)}_{\text{Label compatibility}} \underbrace{w_{ij}(\mathbf{x})}_{\text{Pixel similarity}}$$

Inference in CRFs

- In general NP-hard
- Variational methods: Approximate complex distribution $p(\boldsymbol{y}|\boldsymbol{x})$ with simple distribution $q(\boldsymbol{y})$
- Mean-field approximation: $q(\boldsymbol{y})$ is independent distribution for each pixel:

$$q(\mathbf{y}) = \prod_i q_i(y_i)$$

- If N pixels and K classes, basically N K-dimensional vectors

Mean field inference

- If we can find best q, can then independently optimize each q_i
- Try to match p with q by minimizing *Kulback-Leibler Divergence*

$$KL(q||p) = \sum_{\mathbf{y}} q(\mathbf{y}) \log p(\mathbf{y}) - \sum_{\mathbf{y}} q(\mathbf{y}) \log q(\mathbf{y})$$

- Iterative process: in each iteration, do coordinate ascent on one $q_i(\cdot)$

Mean field inference

- Coordinate descent on $q_i(\cdot)$
- At each step, keep other pixels fixed and update one
- Each step (approximately):
  - Take current $q_j(\cdot)$ on all $j \neq i$
  - Use this to compute $p(y_i|y_{-i})$ where $y_{-i} = \{y_j : j \neq i\}$
    - Easy: only depends on pixels i is connected to
  - Set q_i to this

$$q_i \propto \mathbb{E}_{q_{-i}}[\log p(y_i|y_{-i})]$$

Fully Connected CRFs

- Typically, only adjacent pixels connected
  - Fewer connections => Easier to optimize
- Dense connectivity: every pixel connected to everything else
- Intractable to optimize except if pairwise potential takes specific form

$$\phi_{ij}^{smooth}(y_i, y_j, \mathbf{x}) = \mu(y_i, y_j) w_{ij}(\mathbf{x})$$

$$w_{ij}(\mathbf{x}) = \sum_m w_m e^{-\|\mathbf{f}_m(i) - \mathbf{f}_m(j)\|^2}$$

Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. Philipp Krahenbuhl, Vladlen Koltun. In *NIPS,* 2011.

Gaussian edge potentials

$$w_{ij}(\mathbf{x}) = \sum_m w_m e^{-\|\mathbf{f}_m(i) - \mathbf{f}_m(j)\|^2}$$

- What should **f** be?
- simple answer: color, position

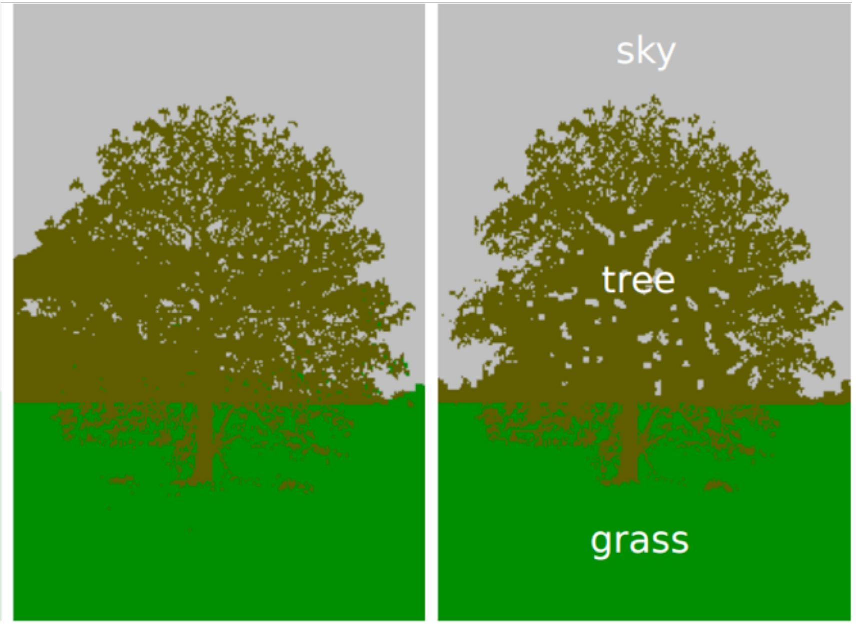
Mean field inference for Dense-CRF

$$q_i(y_i = l) \propto \exp[\underbrace{-\psi_u(y_i)}_{\text{Unary}} - \sum_{l'} \underbrace{\mu(l, l')}_{\text{Label compatibility transform}} \underbrace{\sum_m w_m \sum_{j \neq i} e^{-\|\mathbf{f}_m(i) - \mathbf{f}_m(j)\|^2} q_j(y_j = l')]}_{\text{Message passing}}$$

$$\mathbf{q}_i \propto \exp[-\boldsymbol{\psi}_u^{(i)} - \boldsymbol{\mu} \sum_j \mathbf{m}_{j \to i}]$$

Fully Connected CRFs

Grid CRF



Fully connected CRF

Ground truth

Fully connected CRFs

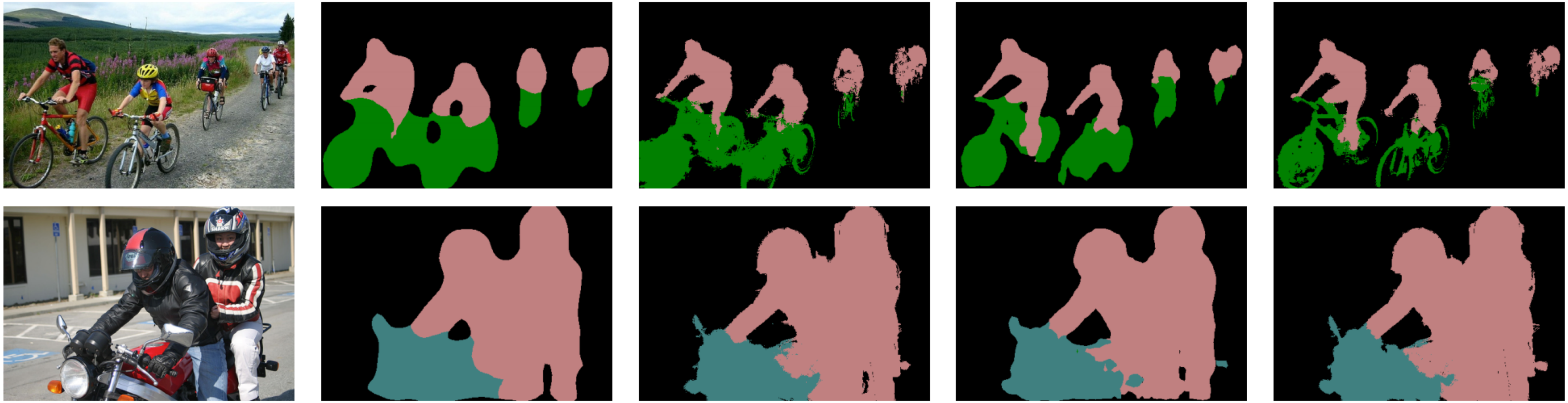


Image VGG-16 Bef. VGG-16 Aft. ResNet Bef. ResNet Aft.

Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, Alan Yuille. In *ICLR,* 2015.

Mean field inference as a recurrent network

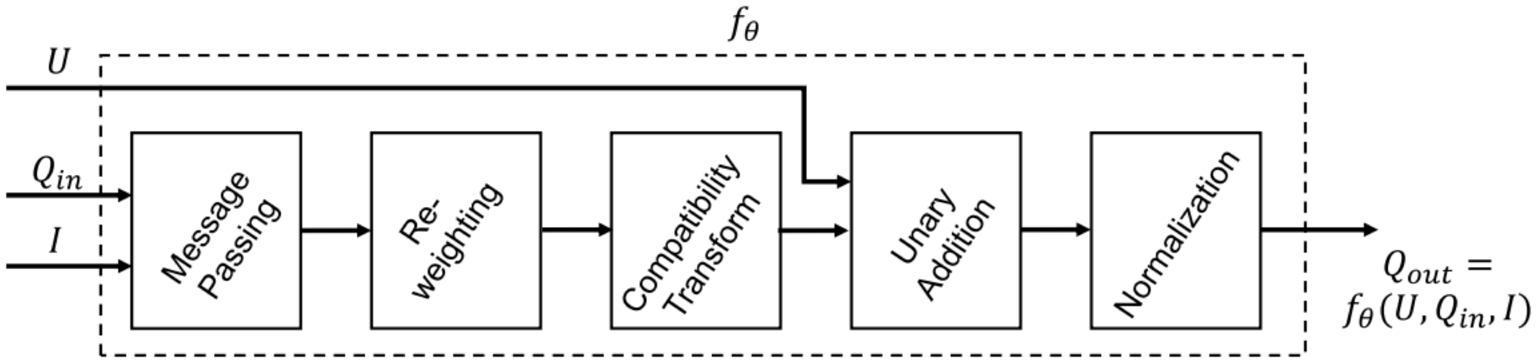


Figure 1. **A mean-field iteration as a CNN.** A single iteration of the mean-field algorithm can be modelled as a stack of common CNN layers.

Zheng, Shuai, et al. "Conditional random fields as recurrent neural networks." *Proceedings of the IEEE international conference on computer vision*. 2015.

Inference as iterated refinement

- Mean field inference and belief propagation are iterative processes
  - In each iteration, update pixel beliefs based on current estimates of neighbors
- In general *approximate* and *may not converge*
- We created ideal probability distribution, then constructed approximate inference scheme
- Can we directly design iterated inference scheme?

Autocontext-like techniques

- Key idea: directly train iterative refinement scheme.
- Initial prediction based just on unaries (for example)
- i-th classifier takes as input image and output of (i-1)-th classifier and makes prediction
- Allows each classifier to use *full context* from previous prediction

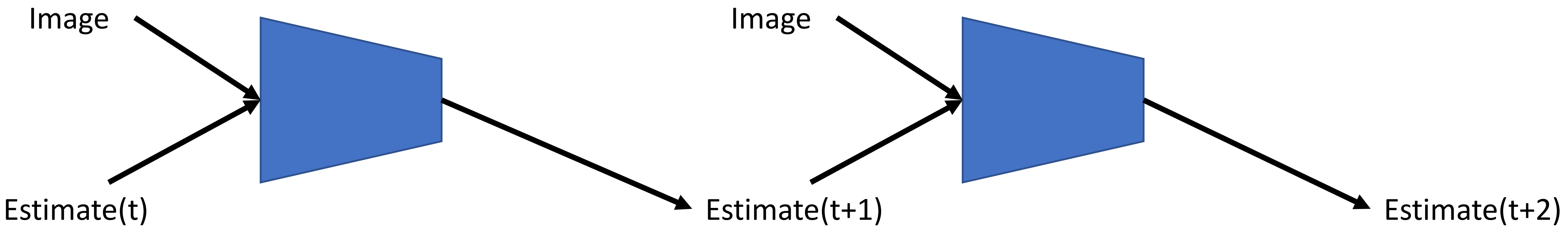
What if i-th classifier overfits?

- *Domain shift* for the next classifier
- Idea: train each classifier on a separate set of inputs
  - Split dataset into 2 folds
  - Train separate classifiers on each fold
  - Use predictions on held-out set to train subsequent predictions

Shared vs separate classifiers

- Using separate classifiers at each step adds parameters
- Can we train a single refinement module?
- *Inference machines*

Autocontext and Inference Machines



- Shared parameters: *Inference Machines*
- Unshared parameters: *Autocontext*

Auto-context and Its Application to High-level Vision Tasks. Zhuowen Tu. In *CVPR* 2008.
Learning Message-Passing Inference Machines for Structured Prediction. Stephane Ross, Daniel Munoz, Martial Hebert, J. Andrew Bagnell. In *CVPR* 2011.