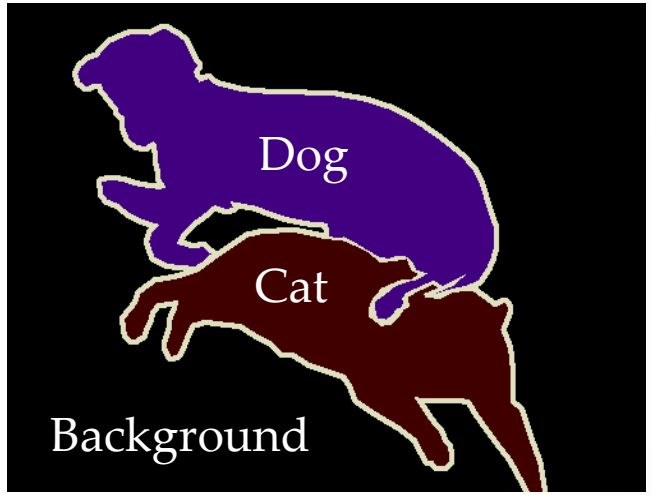
Gaussian Conditional Random Field Network for Semantic Segmentation

Raviteja Vemulapalli, Rama Chellappa
University of Maryland, College Park

Oncel Tuzel, Ming-Yu Liu
Mitsubishi Electric Research Laboratories

Semantic Image Segmentation

- Assign a class label to each pixel in the image.



Deep Neural Networks

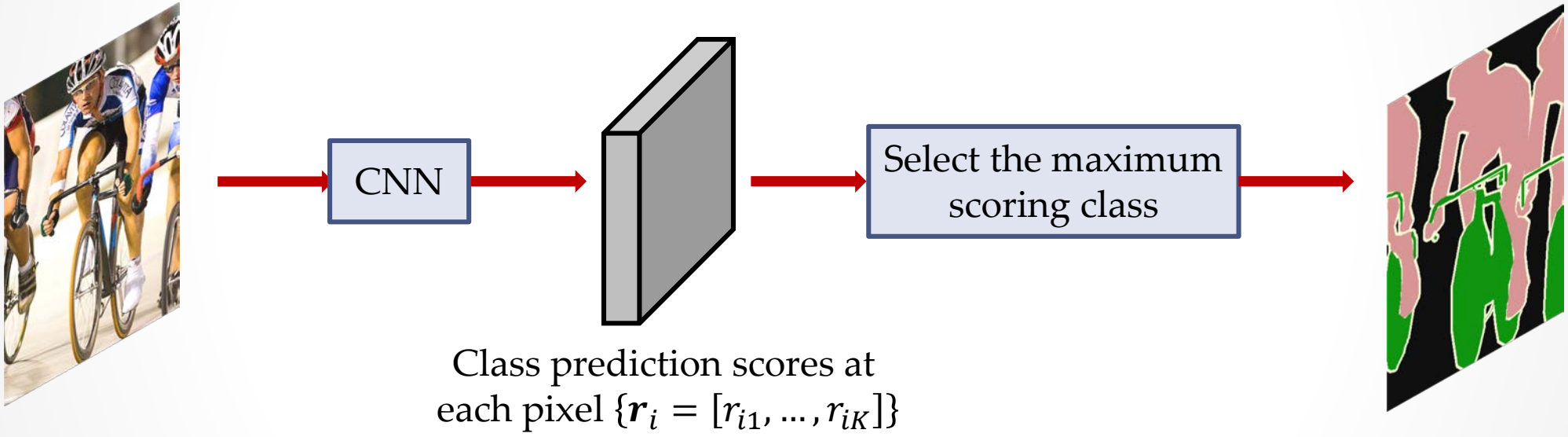
- Deep neural networks have been successfully used in various image processing and computer vision applications:
  - Image denoising, deconvolution and super-resolution
  - Depth estimation
  - Object detection and recognition
  - Semantic segmentation
  - Action recognition

- Their success can be attributed to several factors:
  - Ability to represent complex input-output relationships
  - Feed-forward nature of their inference (no need to solve an optimization problem during run time)
  - Availability of large training datasets and fast computing hardware like GPUs

What is missing in these standard deep neural networks?

CNN-based Semantic Segmentation

- Standard deep networks do not explicitly model the interactions between output variables.



- Modeling the interactions between output variables is very important for structured prediction tasks such as semantic segmentation.

CNN + Discrete CRF

- CRF as a post-processing step

C. Farabet, C. Couprie, L. Najman, and Y. LeCun. *Learning Hierarchical Features for Scene Labeling*. IEEE Trans. Pattern Anal. Mach. Intell., 35(8):1915–1929, 2013.

S. Bell, P. Upchurch, N. Snavely, and K. Bala. *Material Recognition in the Wild with the Materials in Context Database*. In CVPR, 2015.

L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. *Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs*. In ICLR, 2015.

- Joint training of CNN and CRF

S. Zheng, S. Jayasumana, B. R.-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. *Conditional Random Fields as Recurrent Neural Networks*. In ICCV, 2015.

Discrete CRF vs Gaussian CRF

- Discrete CRF is a natural fit for discrete labeling tasks such as semantic segmentation.
- Efficient mean field inference procedure proposed in [Krahenbuhl 2011].
- Inference procedure does not have optimality guarantees.
- For Gaussian CRF, mean field inference gives optimal solution when it converges.
- Not clear if Gaussian CRF is a good fit for discrete labeling tasks.

Should we use a better model with approximate inference
or an approximate model with better inference?

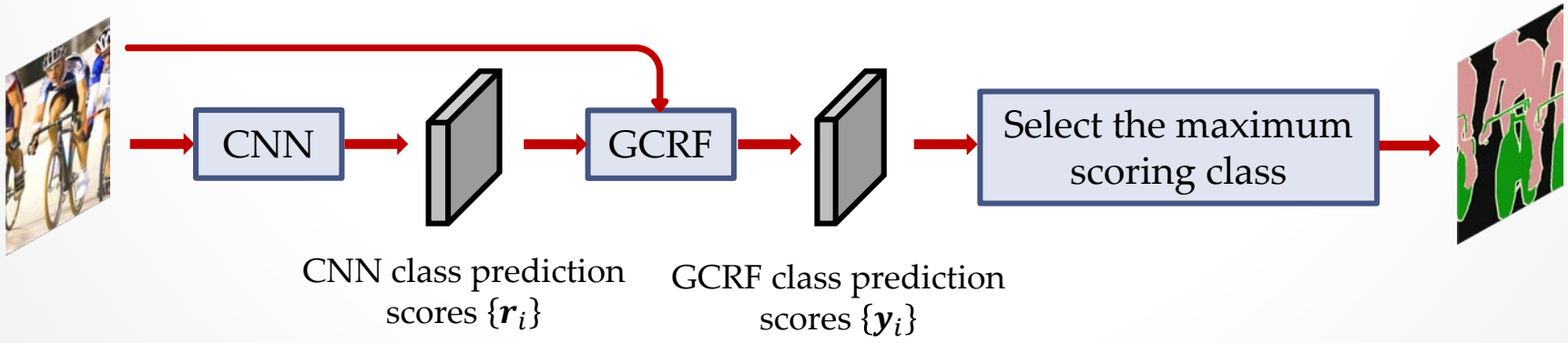
P. Krahenbuhl and V. Koltun, "Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials", NIPS, 2011.

Gaussian CRF for Semantic Segmentation

- We use a Gaussian CRF model on top of a CNN to explicitly model the interactions between the class labels at different pixels.
- Semantic segmentation is a discrete labeling task.
- To use a Gaussian CRF model, we replace each discrete output variable with a vector of K continuous variables:

$$\boldsymbol{y}_i = [y_{i1}, \ldots, y_{iK}] \in R^K.$$

  - y_{ik} represents the score for k^{th} class at i^{th} pixel.
  - Class label for i^{th} pixel is given by $\underset{k}{\operatorname{argmax}}\, y_{ik}$.

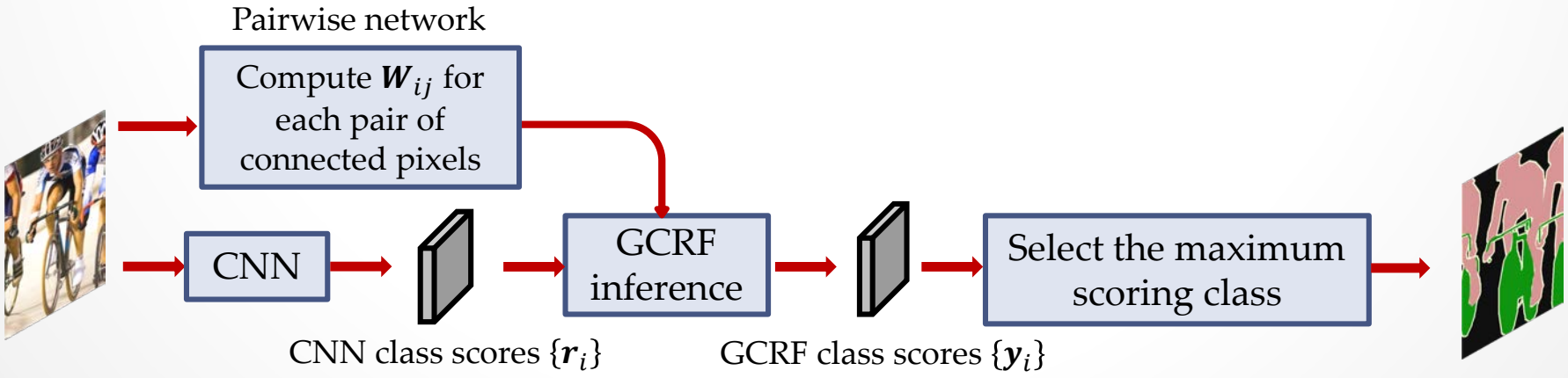


Gaussian CRF Model for Semantic Segmentation

- Let $\boldsymbol{X}$ represent the input image, and $\boldsymbol{Y}$ represent the output (a K-dimensional vector at each pixel).
- We model the conditional probability density $P(\boldsymbol{Y}|\boldsymbol{X})$ as a Gaussian distribution given by $P(\boldsymbol{Y}|\boldsymbol{X}) \propto e^{-\frac{1}{2}E(\boldsymbol{Y}|\boldsymbol{X})}$, where

$$E(\boldsymbol{Y}|\boldsymbol{X}) = \underbrace{\sum_i \|\boldsymbol{y}_i - \boldsymbol{r}_i(\boldsymbol{X};\theta_u)\|_2^2}_{E_u} + \underbrace{\sum_{ij} (\boldsymbol{y}_i - \boldsymbol{y}_j)^T \boldsymbol{W}_{ij}(\boldsymbol{X};\theta_p)(\boldsymbol{y}_i - \boldsymbol{y}_j)}_{E_p}; \quad \boldsymbol{W}_{ij} \geqslant 0.$$

  - $\boldsymbol{r}_i(\boldsymbol{X};\theta_u)$ are the CNN class prediction scores, θ_u are the unary-CNN parameters.
  - $\boldsymbol{W}_{ij}(\boldsymbol{X};\theta_p)$ are the input-dependent parameters of the pairwise potential function E_p.



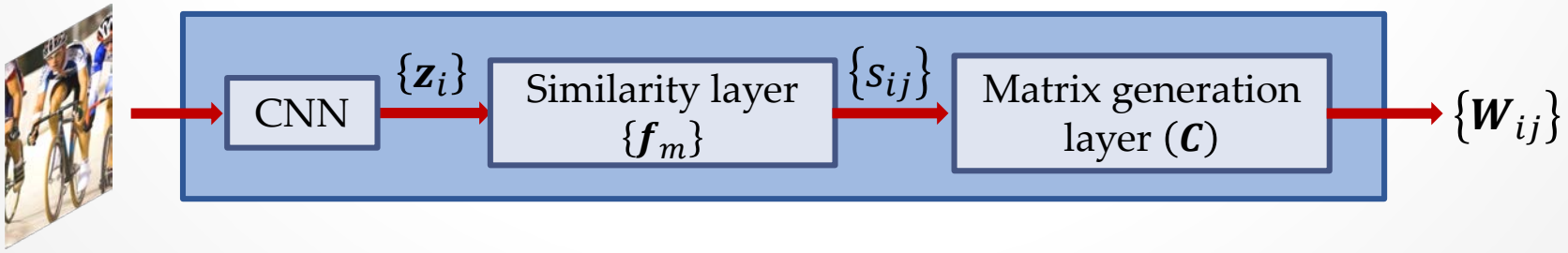
Pairwise Network

- We compute each $\boldsymbol{W}_{ij}$ as $\boldsymbol{W}_{ij} = s_{ij}\boldsymbol{C}; \boldsymbol{C} \succcurlyeq 0$:
  - $s_{ij} \in [0,1]$ is a similarity measure between pixels i and j.
  - $\boldsymbol{C}$ is a parameter matrix that encodes the class compatibility information.

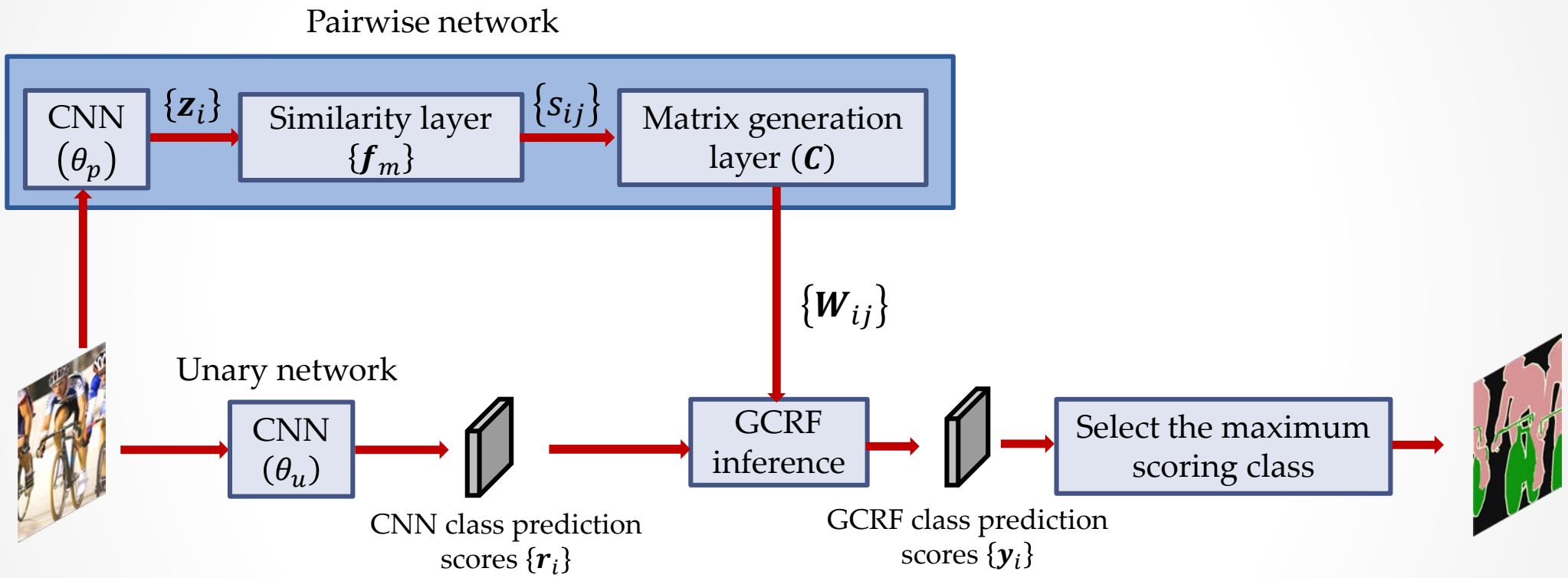
- The similarity measure s_{ij} is computed as $s_{ij} = e^{-(\boldsymbol{z}_i - \boldsymbol{z}_j)^T \boldsymbol{F} (\boldsymbol{z}_i - \boldsymbol{z}_j)}$:
  - $\boldsymbol{z}_i$ is a feature vector extracted at pixel i using a CNN.
  - $\boldsymbol{F} \succcurlyeq 0$ is a parameter matrix that defines a Mahalanobis distance function.

- We implement Mahalanobis distance computation as convolutions followed by Euclidean distance computation.

$$(\boldsymbol{z}_i - \boldsymbol{z}_j)^T \boldsymbol{F} (\boldsymbol{z}_i - \boldsymbol{z}_j) = \sum_{m=1}^{M} (\boldsymbol{f}_m^T \boldsymbol{z}_i - \boldsymbol{f}_m^T \boldsymbol{z}_j)^2; \quad \boldsymbol{F} = \sum_{m=1}^{M} \boldsymbol{f}_m \boldsymbol{f}_m^T.$$



Gaussian CRF Network



GCRF Inference

- Given the unary network output $\{\boldsymbol{r}_i\}$ and the pairwise network output $\{\boldsymbol{W}_{ij}\}$, GCRF inference solves the following optimization problem:

$$\boldsymbol{Y}^* = \underset{\boldsymbol{Y}}{\operatorname{argmax}}\ P(\boldsymbol{Y}|\boldsymbol{X}) = \underset{\boldsymbol{Y}}{\operatorname{argmin}} \sum_i \|\boldsymbol{y}_i - \boldsymbol{r}_i\|_2^2 + \sum_{ij} (\boldsymbol{y}_i - \boldsymbol{y}_j)^T \boldsymbol{W}_{ij} (\boldsymbol{y}_i - \boldsymbol{y}_j).$$

  - Unconstrained quadratic program and hence be solved in closed form.
  - Closed form solution requires solving a linear system with number of variables equal to the number of pixels times the number of classes.

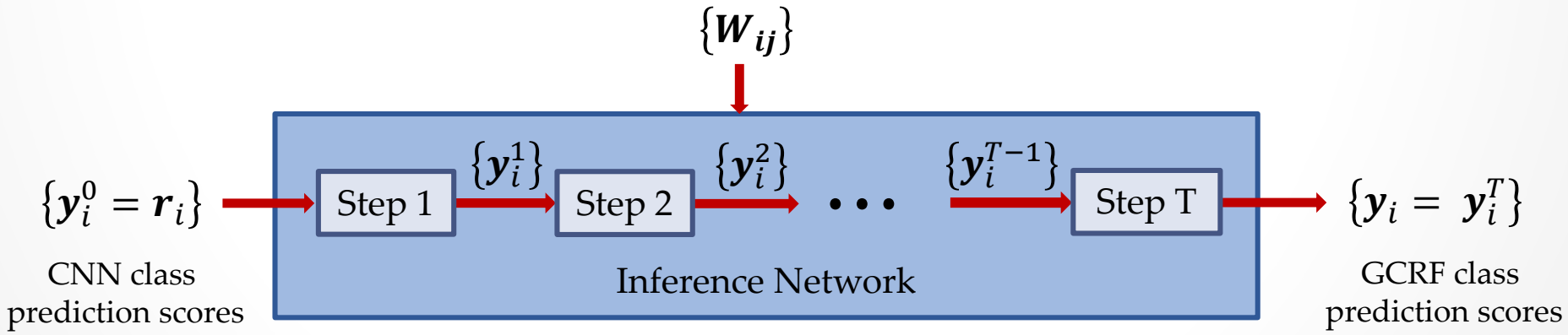
- Instead of exactly solving the full linear system, we perform approximate inference using the iterative Gaussian mean field procedure.

Gaussian Mean Field Inference

- We unroll the iterative Gaussian mean field (GMF) inference into a deep network.

- Parallel GMF inference: Update all the variables in parallel using

$$\boldsymbol{y}_i^{t+1} = \left(I + \sum_j \boldsymbol{W}_{ij}\right)^{-1} \left(\boldsymbol{r}_i + \sum_j \boldsymbol{W}_{ij}\boldsymbol{y}_j^t\right).$$



Convergence of GMF Inference

- Parallel GMF inference is guaranteed to converge to the global optimum if the precision matrix of the Gaussian distribution $P(\boldsymbol{Y}|\boldsymbol{X})$ is diagonal dominant.

$$E(\boldsymbol{Y}|\boldsymbol{X}) = \sum_i \|\boldsymbol{y}_i - \boldsymbol{r}_i\|_2^2 + \sum_{ij} (\boldsymbol{y}_i - \boldsymbol{y}_j)^T \boldsymbol{W}_{ij} (\boldsymbol{y}_i - \boldsymbol{y}_j).$$

- Imposing such constraints on $P(\boldsymbol{Y}|\boldsymbol{X})$ is difficult and could restrict the model capacity in practice.

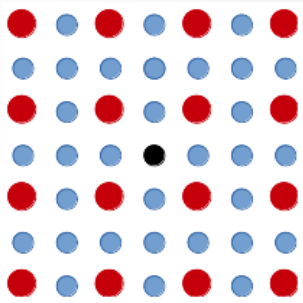
- If we update the variables serially, then GMF inference will converge to the global optimum even without the diagonal dominance constraints.

$$\boldsymbol{y}_i^{t+1} = \left(I + \sum_j \boldsymbol{W}_{ij}\right)^{-1} \left(\boldsymbol{r}_i + \sum_j \boldsymbol{W}_{ij} \boldsymbol{y}_j^t\right).$$

- But serial updates are not practical since we have a huge number of variables.

Convergence of GMF Inference

- Ideally we want to
  - update as many variables as possible in parallel
  - avoid diagonal dominance constraints
  - have convergence guarantee



- When using graphical models, each pixel is usually connected to every pixel within a spatial neighborhood.

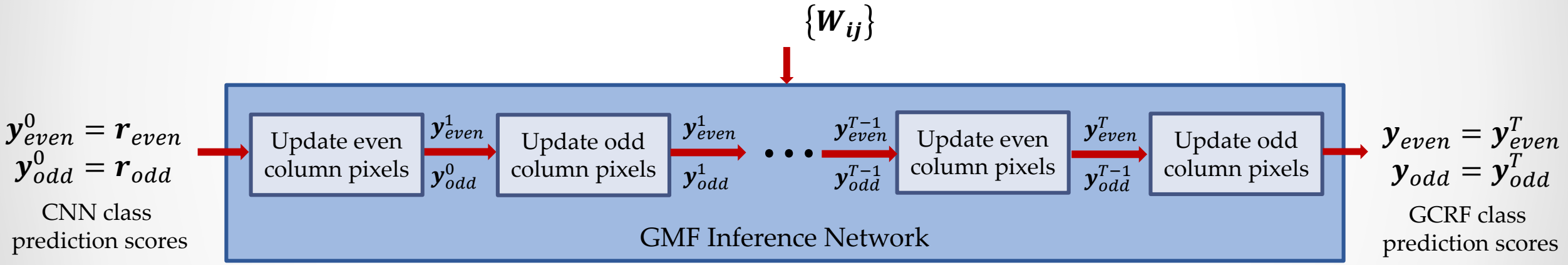
- We connect each pixel to every other pixel along both rows and columns within a spatial neighborhood.

- If we partition the image into even and odd columns, this connectivity ensures that there are no edges within the partitions.

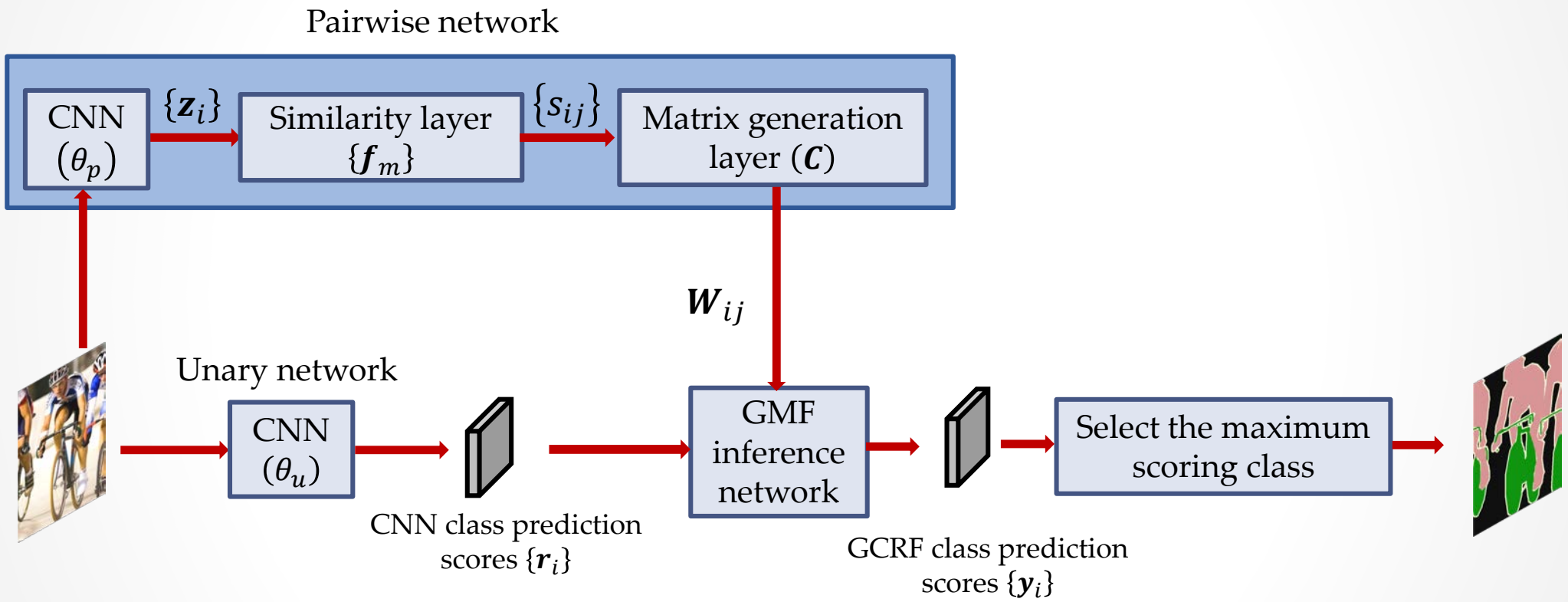
- We can update all even column pixels in parallel and all the odd column pixels in parallel and still have convergence guarantee without the diagonal dominance constraints.

GMF Inference Network

- Each layer of our network produces an output that is closer to the optimal solution compared to its input (unless the input itself is the optimal solution, in which case the output will be equal to the input).



GCRF Network



Training

- CNNs were initialized using DeepLab CNN model.
- Pairwise network pre-trained like a Siamese network at pixel level.
- Trained the GCRF network end-to-end discriminatively.
- Training loss function:

$$L(\{\boldsymbol{y}_i, l_i\}) = -\frac{1}{N}\sum_{i=1}^{N} \min\left(0,\ y_{il_i} - \max_{k \neq l_i} y_{ik} - S\right).$$

  - l_i is the true class label of pixel i.
  - This cost function encourages the prediction score for the true class to be greater than the prediction scores of all the other classes by a margin S.
- Used standard back-propagation to compute the gradient of the network parameters.
- We have a constrained optimization because of the symmetry and positive semi-definiteness constraints on the parameter matrix $\boldsymbol{C}$.
- Parametrized $\boldsymbol{C}$ as $\boldsymbol{C} = \boldsymbol{R}\boldsymbol{R}^{\boldsymbol{T}}$ where $\boldsymbol{R}$ is a lower triangular matrix, and used stochastic gradient descent.

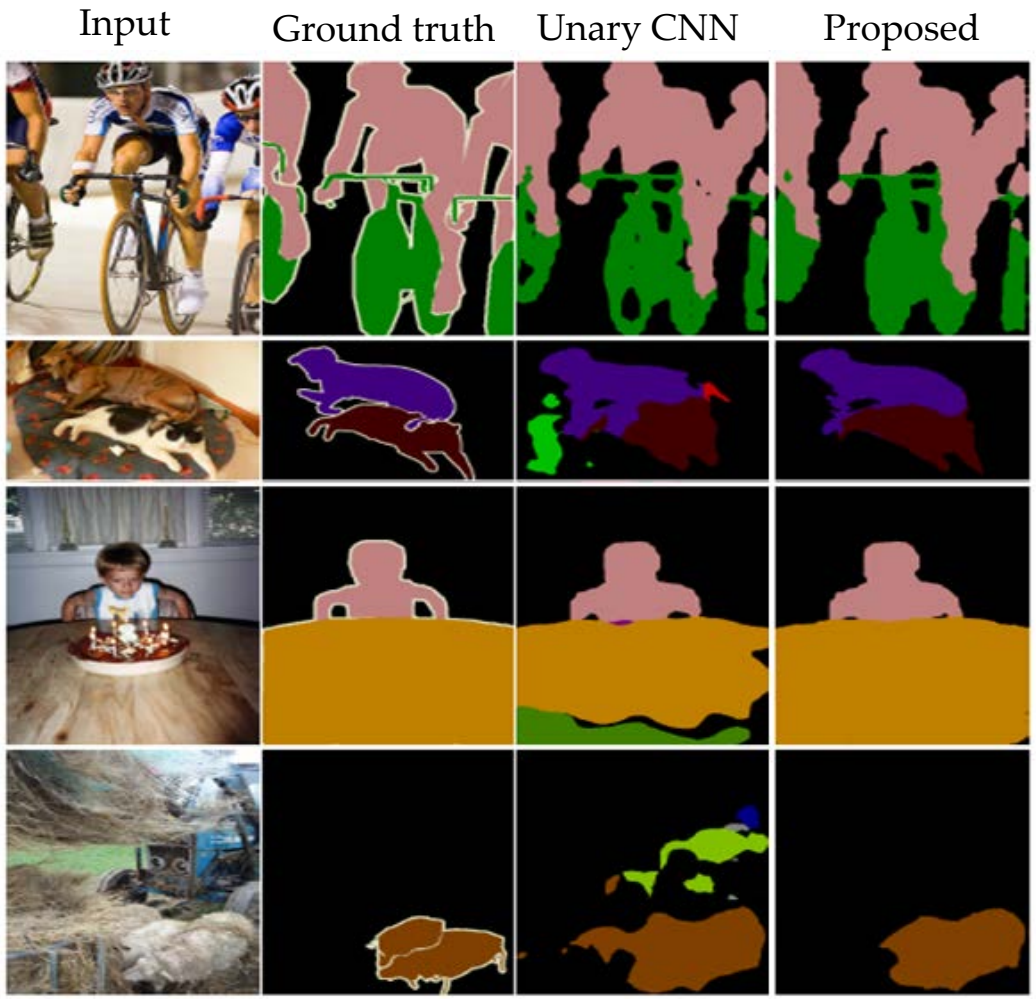
Experimental Results

- PASCALVOC2012 dataset: 10,582 training images and 1456 test images.

- Mean IOU score: 73.2 (better than the unary CNN by 6.2 points)

Method	bkg	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbk	person	plant	sheep	sofa	train	tv	mean
MSRA-CFM [9]	87.7	75.7	26.7	69.5	48.8	65.6	81.0	69.2	73.3	30.0	68.7	51.5	69.1	68.1	71.7	67.5	50.4	66.5	44.4	58.9	53.5	61.8
FCN-8s [29]	91.2	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
Hypercolumns [18]	89.3	68.7	33.5	69.8	51.3	70.2	81.1	71.9	74.9	23.9	60.6	46.9	72.1	68.3	74.5	72.9	52.6	64.4	45.4	64.9	57.4	62.6
DeepLab CNN [7]	91.6	78.7	51.5	75.8	59.5	61.9	82.5	76.6	79.4	26.9	67.7	54.7	74.3	70.0	79.8	77.3	52.6	75.2	46.6	66.9	57.3	67.0
ZoomOut [30]	91.1	85.6	37.3	83.2	62.5	66.0	85.1	80.7	84.9	27.2	73.2	57.5	78.1	79.2	81.1	77.1	53.6	74.0	49.2	71.7	63.3	69.6
Deep message passing [26]	93.9	90.1	38.6	77.8	61.3	74.3	89.0	83.4	83.3	36.2	80.2	56.4	81.2	81.4	83.1	82.9	59.2	83.4	54.3	80.6	70.8	73.4
Approaches that use CNNs and discrete CRFs																						
Deep structure models [27]	93.6	86.7	36.9	82.3	63.0	74.2	89.8	84.1	84.1	32.8	65.4	52.1	79.7	72.1	77.6	81.7	55.6	77.4	37.4	81.4	68.4	70.3
DeconvNet + CRF [31]	92.9	87.8	41.9	80.6	63.9	67.3	88.1	78.4	81.3	25.9	73.7	61.2	72.0	77.0	79.9	78.7	59.5	78.3	55.0	75.2	61.5	70.5
object clique potentials [36]	92.8	80.0	53.8	80.8	62.5	64.7	87.0	78.5	83.0	29.0	82.0	60.3	76.3	78.4	83.0	79.8	57.0	80.0	53.1	70.1	63.1	71.2
DeepLab CNN-CRF [7]	93.3	84.4	54.5	81.5	63.6	65.9	85.1	79.1	83.4	30.7	74.1	59.8	79.0	76.1	83.2	80.8	59.7	82.2	50.4	73.1	63.7	71.6
CRF-RNN [54]	94.0	87.5	39.0	79.7	64.2	68.3	87.6	80.8	84.4	30.4	78.2	60.4	80.5	77.8	83.1	80.6	59.5	82.8	47.8	78.3	67.1	72.0
DeconvNet + FCN + CRF [31]	93.1	89.9	39.3	79.7	63.9	68.2	87.4	81.2	86.1	28.5	77.0	62.0	79.0	80.3	83.6	80.2	58.8	83.4	54.3	80.7	65.0	72.5
Proposed GCRF network	93.4	85.2	43.9	83.3	65.2	68.3	89.0	82.7	85.3	31.1	79.5	63.3	80.5	79.3	85.5	81.0	60.5	85.5	52.0	77.3	65.1	73.2

Experimental Results



Experimental Results

Thank You