Machine Learning

A. G. Schwing & M. Telgarsky

University of Illinois at Urbana-Champaign, 2018

L11: Structured Prediction (exhaustive search, dynamic programming)

Goals of this lecture

Goals of this lecture

- Getting to know structured prediction

Goals of this lecture

- Getting to know structured prediction
- Understanding basic structured inference algorithms

Goals of this lecture

- Getting to know structured prediction
- Understanding basic structured inference algorithms

Reading material:

Goals of this lecture

- Getting to know structured prediction
- Understanding basic structured inference algorithms

Reading material:

- D. Koller and N. Friedman; Probabilistic Graphical Models: Principles and Techniques;

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)},\hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

How to get to

- Logistic regression

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

How to get to

- Logistic regression
- Binary SVM

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

How to get to

- Logistic regression
- Binary SVM
- Multiclass regression

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)},\hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

How to get to

- Logistic regression
- Binary SVM
- Multiclass regression
- Multiclass SVM

Recap: General framework for learning:

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + F(\boldsymbol{w}, x^{(i)}, \hat{y})}{\epsilon} - F(\boldsymbol{w}, x^{(i)}, y^{(i)})$$

Attention:

- Scoring function $F(\boldsymbol{w}, x^{(i)}, y^{(i)})$
- Loss function (log-loss, hinge-loss)
- Taskloss L

How to get to

- Logistic regression
- Binary SVM
- Multiclass regression
- Multiclass SVM
- Deep Learning

Recap: Inference (how to find the highest scoring configuration):

Recap: Inference (how to find the highest scoring configuration):

$$y^* = \arg\max_{\hat{y}} F(\boldsymbol{w}, x, \hat{y})$$

Recap: Inference (how to find the highest scoring configuration):

$$y^* = \arg\max_{\hat{y}} F(\boldsymbol{w}, x, \hat{y})$$

How to solve it?

Recap: Inference (how to find the highest scoring configuration):

$$y^* = \arg\max_{\hat{y}} F(\boldsymbol{w}, x, \hat{y})$$

How to solve it?

Exhaustive search over all classes (easy for binary/few classes)

Example: why is structure/are correlations useful

Example: why is structure/are correlations useful

Example: why is structure/are correlations useful

V

Example: why is structure/are correlations useful

Example: why is structure/are correlations useful

I

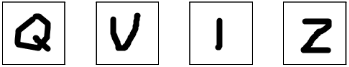
Example: why is structure/are correlations useful

Example: why is structure/are correlations useful

Z

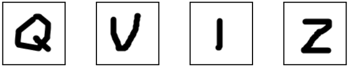
Example: why is structure/are correlations useful

Example: why is structure/are correlations useful

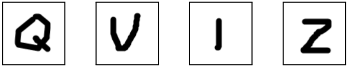


Q V I Z

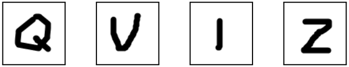
Example: why is structure/are correlations useful



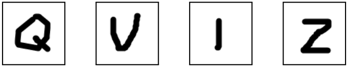
Q V I Z



Example: why is structure/are correlations useful

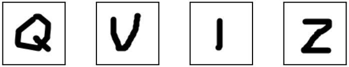


Q V I Z

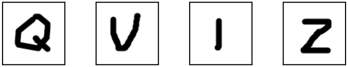


Q U I Z

Example: why is structure/are correlations useful



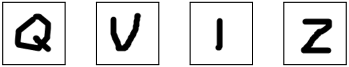
Q V I Z



Q U I Z

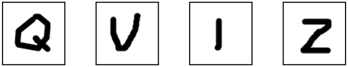
Correlations not taken into account

Example: why is structure/are correlations useful



Q V I Z

Correlations not taken into account



Q U I Z

Correlations taken into account

How can we take correlations into account?

How can we take correlations into account?

- Formulate it as prediction of all four letter words (multiclass prediction):

How can we take correlations into account?

- Formulate it as prediction of all four letter words (multiclass prediction):

$$y \in \mathcal{Y} = \{1, \ldots, 26^4\}$$

How can we take correlations into account?

- Formulate it as prediction of all four letter words (multiclass prediction):

$$y \in \mathcal{Y} = \{1, \ldots, 26^4\}$$

- Problem:

How can we take correlations into account?

- Formulate it as prediction of all four letter words (multiclass prediction):

$$y \in \mathcal{Y} = \{1, \ldots, 26^4\}$$

- Problem: Really large **output space**

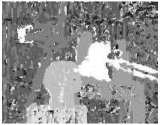
Example: Disparity map estimation

How big is the output space?

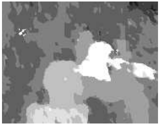
Example: Disparity map estimation

How big is the output space?

Image



Independent Prediction



Structured Prediction

Example: De-noising

Example: De-noising

$\rightarrow$

Example: De-noising

$\rightarrow$

Predictions from neighboring pixel are useful.

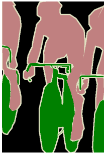
Structured Prediction estimates a complex object

$$x^{(i)} \quad \rightarrow \quad \boldsymbol{y}^{(i)} = (y_1^{(i)}, \ldots, y_D^{(i)})$$

Structured Prediction estimates a complex object

- Image segmentation (estimate a labeling)

$x^{(i)}$ $\rightarrow$ $\boldsymbol{y}^{(i)} = (y_1^{(i)}, \ldots, y_D^{(i)})$

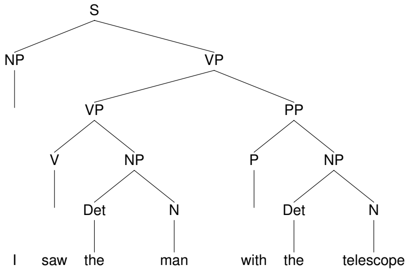


Structured Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)

$$x^{(i)} \quad \rightarrow \quad \boldsymbol{y}^{(i)} = (y_1^{(i)}, \ldots, y_D^{(i)})$$

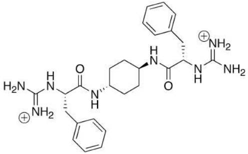
I saw the man with the telescope.

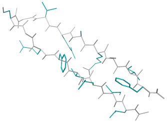


Structured Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)
- Protein folding (estimate a protein structure)

$$x^{(i)} \quad \rightarrow \quad \mathbf{y}^{(i)} = (y_1^{(i)}, \ldots, y_D^{(i)})$$

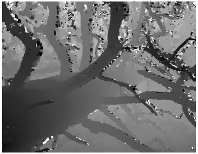




Structured Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)
- Protein folding (estimate a protein structure)
- Stereo vision (estimate a disparity map)

$$x^{(i)} \quad \rightarrow \quad \boldsymbol{y}^{(i)} = (y_1^{(i)}, \ldots, y_D^{(i)})$$



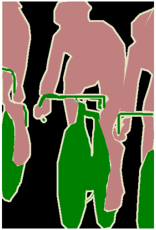
- “Standard” Prediction: output $y \in \mathcal{Y}$ is a scalar number

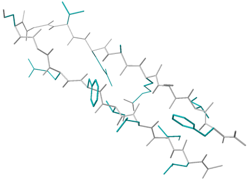
$$\mathcal{Y} = \{1, \ldots, K\} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}$$

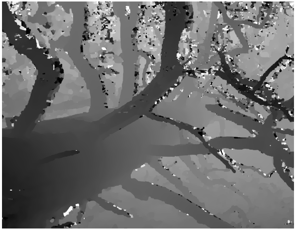
- "Standard" Prediction: output $y \in \mathcal{Y}$ is a scalar number

$$\mathcal{Y} = \{1, \ldots, K\} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}$$

- "Structured" Prediction: output $\boldsymbol{y}$ is a structured output:



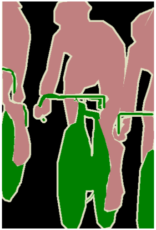


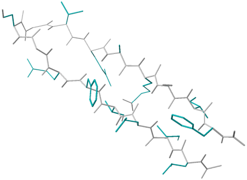


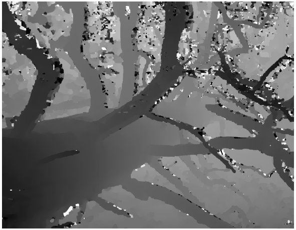
- "Standard" Prediction: output $y \in \mathcal{Y}$ is a scalar number

$$\mathcal{Y} = \{1, \ldots, K\} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}$$

- "Structured" Prediction: output $\boldsymbol{y}$ is a structured output:







We can transition between both formulations. K possibly really large.

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Inference/Prediction:

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Inference/Prediction:

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) =$$

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Inference/Prediction:

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D)$$

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Inference/Prediction:

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D)$$

How many possibilities do we have to store and explore?

Formally:

$$\boldsymbol{y} = (y_1, \ldots, y_D) \qquad y_d \in \{1, \ldots, K\}$$

Inference/Prediction:

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \arg\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D)$$

How many possibilities do we have to store and explore?

$$K^D$$

That's a problem. What can we do?

Separate prediction:

If

$$F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \max_{\hat{\boldsymbol{y}}} \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d) =$$

Separate prediction:

If

$$F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \max_{\hat{\boldsymbol{y}}} \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d) = \sum_{d=1}^{D} \max_{\hat{y}_d} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

Separate prediction:

If

$$F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \max_{\hat{\boldsymbol{y}}} \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d) = \sum_{d=1}^{D} \max_{\hat{y}_d} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

Why not predict every variable y_d from $\boldsymbol{y} = (y_1, \ldots, y_D)$ separately?

Separate prediction:

If

$$F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{y}_1, \ldots, \hat{y}_D) = \max_{\hat{\boldsymbol{y}}} \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, \hat{y}_d) = \sum_{d=1}^{D} \max_{\hat{y}_d} f_d(\boldsymbol{w}, x, \hat{y}_d)$$

Why not predict every variable y_d from $\boldsymbol{y} = (y_1, \ldots, y_D)$ separately?

Correlations not explicitly taken into account

Score function decomposes less trivially:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{r \in \mathcal{R}} f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

Restriction set $r \subseteq \{1, \ldots, D\}$
Set of all restrictions: $\mathcal{R}$

Score function decomposes less trivially:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{r \in \mathcal{R}} f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

Restriction set $r \subseteq \{1, \ldots, D\}$
Set of all restrictions: $\mathcal{R}$
Example: $r = \{1, 2\}$

$$f_{\{1,2\}}(\boldsymbol{y}_{\{1,2\}}) = f_{\{1,2\}}(y_1, y_2) = \left[f_{\{1,2\}}(1,1), f_{\{1,2\}}(1,2), \ldots\right]$$

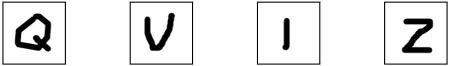
Score function decomposes less trivially:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{r \in \mathcal{R}} f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

Restriction set $r \subseteq \{1, \ldots, D\}$
Set of all restrictions: $\mathcal{R}$
Example: $r = \{1, 2\}$

$$f_{\{1,2\}}(\boldsymbol{y}_{\{1,2\}}) = f_{\{1,2\}}(y_1, y_2) = \left[f_{\{1,2\}}(1,1), f_{\{1,2\}}(1,2), \ldots \right]$$

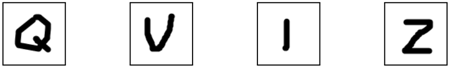
	Q	U	I	Z	V
Q	0	0.8	0.2	0.1	0.1
U		$\ddots$	$\cdots$		
I	$\vdots$				
Z					
V					



Q U I Z

Example:

$$F(\mathbf{w}, x, y_1, \ldots, y_4) = f_1(\mathbf{w}, x, y_1) + f_2(\mathbf{w}, x, y_2) + f_3(\mathbf{w}, x, y_3) + f_4(\mathbf{w}, x, y_4)$$
$$+ f_{1,2}(\mathbf{w}, x, y_1, y_2) + f_{2,3}(\mathbf{w}, x, y_2, y_3) + f_{3,4}(\mathbf{w}, x, y_3, y_4)$$

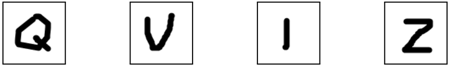


Q U I Z

Example:

$$F(\mathbf{w}, x, y_1, \ldots, y_4) = f_1(\mathbf{w}, x, y_1) + f_2(\mathbf{w}, x, y_2) + f_3(\mathbf{w}, x, y_3) + f_4(\mathbf{w}, x, y_4) + f_{1,2}(\mathbf{w}, x, y_1, y_2) + f_{2,3}(\mathbf{w}, x, y_2, y_3) + f_{3,4}(\mathbf{w}, x, y_3, y_4)$$

How many function values need to be stored if $y_d \in \{1, \ldots, 26\}\ \forall d$?



Q U I Z

Example:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_4) = f_1(\boldsymbol{w}, x, y_1) + f_2(\boldsymbol{w}, x, y_2) + f_3(\boldsymbol{w}, x, y_3) + f_4(\boldsymbol{w}, x, y_4) + f_{1,2}(\boldsymbol{w}, x, y_1, y_2) + f_{2,3}(\boldsymbol{w}, x, y_2, y_3) + f_{3,4}(\boldsymbol{w}, x, y_3, y_4)$$

How many function values need to be stored if $y_d \in \{1, \ldots, 26\}\ \forall d$?

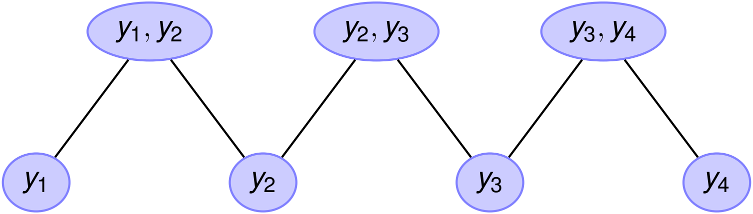
Earlier: 26^4 v.s. now: $3 \cdot 26^2 (+4 \cdot 26)$

Visualization of the decomposition:

$$\begin{aligned} F(\boldsymbol{w}, x, y_1, \ldots, y_4) = & f_1(\boldsymbol{w}, x, y_1) + f_2(\boldsymbol{w}, x, y_2) + f_3(\boldsymbol{w}, x, y_3) + f_4(\boldsymbol{w}, x, y_4) \\ & + f_{1,2}(\boldsymbol{w}, x, y_1, y_2) + f_{2,3}(\boldsymbol{w}, x, y_2, y_3) + f_{3,4}(\boldsymbol{w}, x, y_3, y_4) \end{aligned}$$

Visualization of the decomposition:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_4) = f_1(\boldsymbol{w}, x, y_1) + f_2(\boldsymbol{w}, x, y_2) + f_3(\boldsymbol{w}, x, y_3) + f_4(\boldsymbol{w}, x, y_4) + f_{1,2}(\boldsymbol{w}, x, y_1, y_2) + f_{2,3}(\boldsymbol{w}, x, y_2, y_3) + f_{3,4}(\boldsymbol{w}, x, y_3, y_4)$$



Edges denote subset relationship

Special cases

- Predicting every variable separately:

Special cases

- Predicting every variable separately:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d)$$

Special cases

- Predicting every variable separately:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d)$$

Markov random field with only unary variables

Special cases

- Predicting every variable separately:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d)$$

Markov random field with only unary variables

- Multi-variate prediction:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = f_{1,\ldots,D}(\boldsymbol{w}, x, \boldsymbol{y}_{1,\ldots,D})$$

Special cases

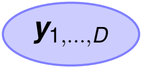
- Predicting every variable separately:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d)$$

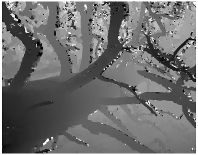
Markov random field with only unary variables

- Multi-variate prediction:

$$F(\boldsymbol{w}, x, y_1, \ldots, y_D) = f_{1,\ldots,D}(\boldsymbol{w}, x, \boldsymbol{y}_{1,\ldots,D})$$

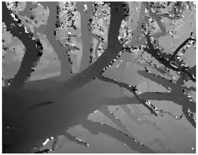


Example: stereo vision

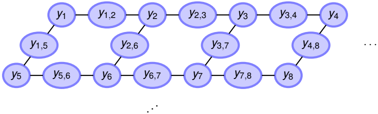


Markov/Conditional random field:

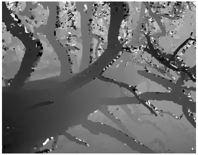
Example: stereo vision



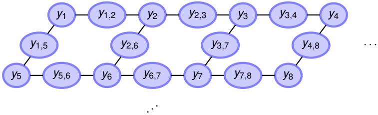
Markov/Conditional random field:



Example: stereo vision

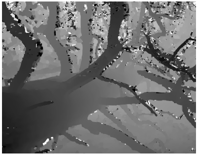


Markov/Conditional random field:

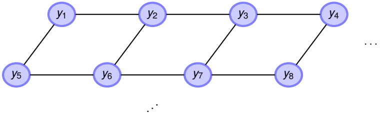


$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

Example: stereo vision

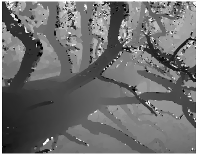


Markov/Conditional random field:

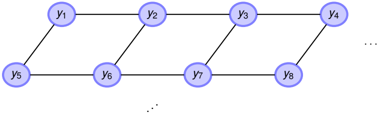


$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

Example: stereo vision



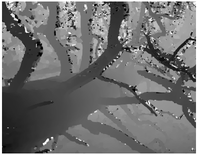
Markov/Conditional random field:



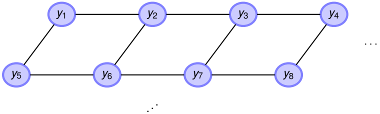
$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

- Unary term:
- Pairwise term:

Example: stereo vision

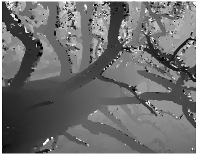
Markov/Conditional random field:



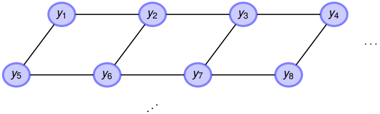
$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

- Unary term: image evidence
- Pairwise term:

Example: stereo vision



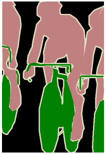
Markov/Conditional random field:



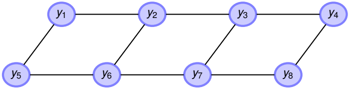
$$F(\mathbf{w}, x, \mathbf{y}) = \sum_{d=1}^{D} f_d(\mathbf{w}, x, y_d) + \sum_{i,j} f_{i,j}(\mathbf{w}, x, y_i, y_j)$$

- Unary term: image evidence
- Pairwise term: smoothness prior

Example: semantic segmentation

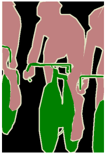


Markov/Conditional random field:

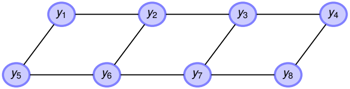


$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

Example: semantic segmentation



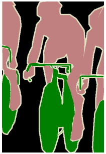
Markov/Conditional random field:



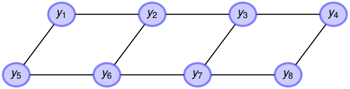
$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

- Unary term:
- Pairwise term:

Example: semantic segmentation



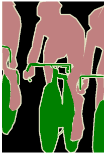
Markov/Conditional random field:



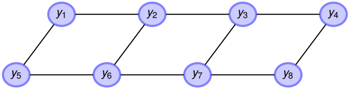
$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

- Unary term: image evidence
- Pairwise term:

Example: semantic segmentation

Markov/Conditional random field:



$$F(\boldsymbol{w}, x, \boldsymbol{y}) = \sum_{d=1}^{D} f_d(\boldsymbol{w}, x, y_d) + \sum_{i,j} f_{i,j}(\boldsymbol{w}, x, y_i, y_j)$$

- Unary term: image evidence
- Pairwise term: smoothness prior

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search
- Dynamic programming

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation
- Message passing

Inference: (how to find the highest scoring configuration)

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation
- Message passing
- Graph-cut

Exhaustive Search

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Exhaustive Search

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Algorithm:

- try all possible configurations $\hat{\boldsymbol{y}} \in \mathcal{Y}$
- keep highest scoring element

Exhaustive Search

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Algorithm:

- try all possible configurations $\hat{\boldsymbol{y}} \in \mathcal{Y}$
- keep highest scoring element

- **Advantage:**
- **Disadvantage:**

Exhaustive Search

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Algorithm:

- try all possible configurations $\hat{\boldsymbol{y}} \in \mathcal{Y}$
- keep highest scoring element

- **Advantage:** very simple to implement
- **Disadvantage:**

Exhaustive Search

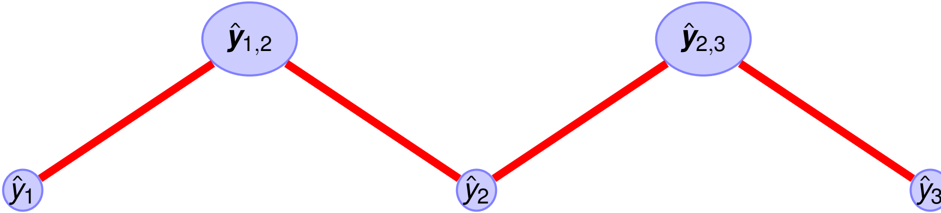
$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}} \sum_r f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$$

Algorithm:

- try all possible configurations $\hat{\boldsymbol{y}} \in \mathcal{Y}$
- keep highest scoring element

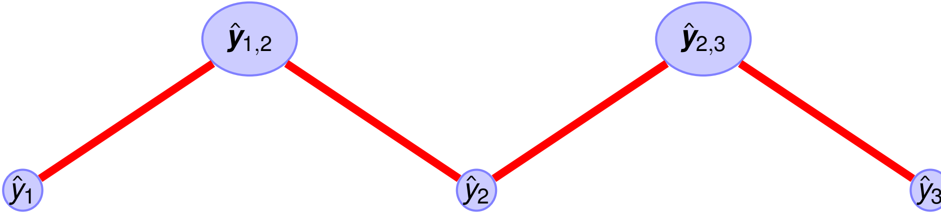
- **Advantage:** very simple to implement
- **Disadvantage:** very slow for reasonably sized problems: K^D

Dynamic Programming



$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \max_{\hat{y}_1, \hat{y}_2, \hat{y}_3} \left(f_3(\hat{y}_3) + f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2)\right)$$

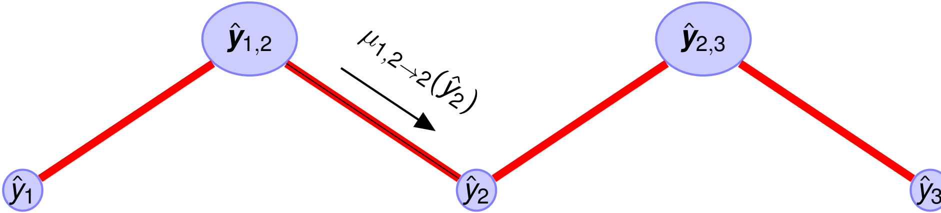
Dynamic Programming



$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \max_{\hat{y}_1, \hat{y}_2, \hat{y}_3} \left(f_3(\hat{y}_3) + f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, y_3) + f_2(\hat{y}_2) + \max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\} \right) \right)$$

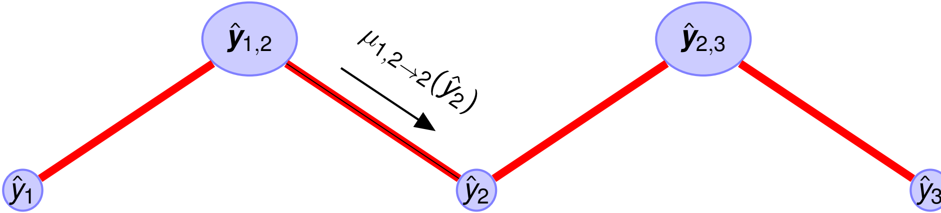
Dynamic Programming



$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \max_{\hat{y}_1,\hat{y}_2,\hat{y}_3} \left(f_3(\hat{y}_3) + f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2)\right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, y_3) + f_2(\hat{y}_2) + \underbrace{\max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\}}_{\mu_{1,2\to 2}(\hat{y}_2)} \right) \right)$$

Dynamic Programming

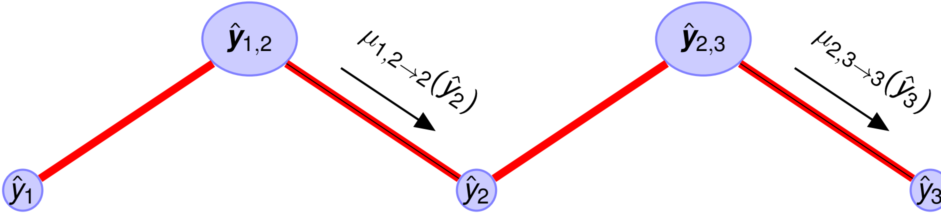


$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \max_{\hat{y}_1,\hat{y}_2,\hat{y}_3} \left(f_3(\hat{y}_3) + f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2)\right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, y_3) + f_2(\hat{y}_2) + \underbrace{\max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\}}_{\mu_{1,2\to 2}(\hat{y}_2)} \right) \right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + \mu_{1,2\to 2}(\hat{y}_2) \right) \right)$$

Dynamic Programming



$$\max_{\hat{\boldsymbol{y}}} F(\boldsymbol{w}, x, \hat{\boldsymbol{y}}) = \max_{\hat{y}_1,\hat{y}_2,\hat{y}_3} \left(f_3(\hat{y}_3) + f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2)\right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, y_3) + f_2(\hat{y}_2) + \underbrace{\max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\}}_{\mu_{1,2\to2}(\hat{y}_2)} \right) \right)$$

$$= \max_{\hat{y}_3} \left(f_3(\hat{y}_3) + \max_{\hat{y}_2} \left(f_{2,3}(\hat{y}_2, \hat{y}_3) + f_2(\hat{y}_2) + \mu_{1,2\to2}(\hat{y}_2) \right) \right)$$

Dynamic Programming

When is this approach suitable:

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

- **Advantage:**
- **Disadvantage:**

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

- **Advantage:** better complexity than exhaustive search: $D \cdot K^2$ for pairwise models
- **Disadvantage:**

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

- **Advantage:** better complexity than exhaustive search: $D \cdot K^2$ for pairwise models
- **Disadvantage:** only works for trees

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

- **Advantage:** better complexity than exhaustive search: $D \cdot K^2$ for pairwise models
- **Disadvantage:** only works for trees

What to do for general loopy graphs?

Dynamic Programming

When is this approach suitable:

We can reorganize terms whenever the graph is a **tree.**

- **Advantage:** better complexity than exhaustive search: $D \cdot K^2$ for pairwise models
- **Disadvantage:** only works for trees

What to do for general loopy graphs?

- Integer Linear Programs
- Linear Programming relaxations
- Dynamic programming extensions (message passing)
- Graph cut algorithms

Quiz:

Quiz:

- Why structured output spaces?

Quiz:

- Why structured output spaces?
- What makes computation with structured spaces hard?

Quiz:

- Why structured output spaces?
- What makes computation with structured spaces hard?
- Inference algorithms for structured output spaces?

Important topics of this lecture

- Getting to know structured prediction
- Understood some inference algorithms

Up next:

- More inference algorithms for structured output spaces