Machine Learning

A. G. Schwing & M. Telgarsky

University of Illinois at Urbana-Champaign, 2018

L13: Conditional Random Fields, Structured SVMs, Deep Structured Nets

Goals of this lecture

Goals of this lecture

- Learning of structured distributions

Goals of this lecture

- Learning of structured distributions

Reading material:

Goals of this lecture

- Learning of structured distributions

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 $F(\boldsymbol{w}, x, y)$

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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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 $F(\boldsymbol{w}, x, y)$
- 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):

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

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

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

Structured Prediction

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

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

Structured Prediction

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

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

- Exhaustive search

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

- Exhaustive search
- Dynamic programming

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

- Exhaustive search
- Dynamic programming
- Integer linear program

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

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

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

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

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

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

Structured Prediction

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

Efficiency and accuracy of inference algorithms is problem dependent:

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

Question for today:

Question for today:

How to put more complex objects $\boldsymbol{y}^{(i)}$ (as opposed to $y^{(i)}$) into our learning formulation?

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

Question for today:

How to put more complex objects $\boldsymbol{y}^{(i)}$ (as opposed to $y^{(i)}$) into our learning formulation?

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

Let's start with our linear multi-class formulation:

Question for today:

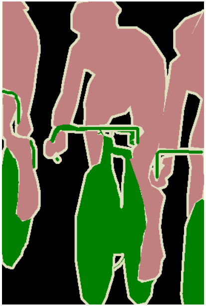
How to put more complex objects $\boldsymbol{y}^{(i)}$ (as opposed to $y^{(i)}$) into our learning formulation?

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

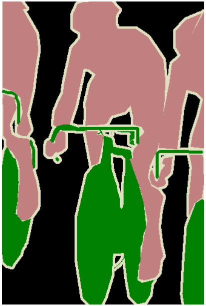
Let's start with our linear multi-class formulation:

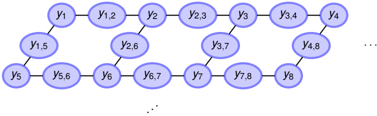
$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{L(y^{(i)}, \hat{y}) + \boldsymbol{w}^T\psi(x^{(i)}, \hat{y}))}{\epsilon} - \boldsymbol{w}^T\psi(x^{(i)}, y^{(i)})$$

Example: Semantic segmentation

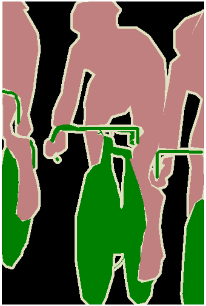


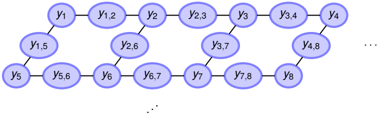
Example: Semantic segmentation





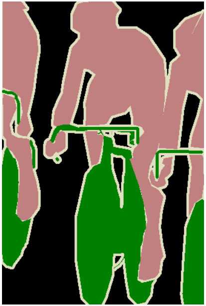
Example: Semantic segmentation

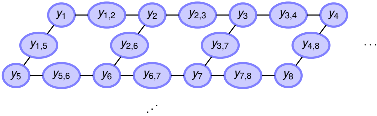




- A classifier provides evidence for every pixel (unary potentials)

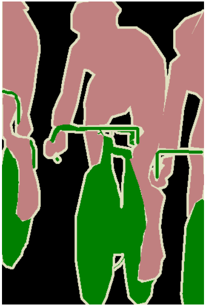
Example: Semantic segmentation

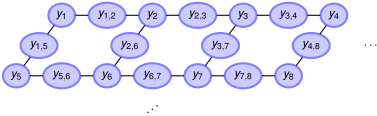




- A classifier provides evidence for every pixel (unary potentials)
- Smoothness is encouraged via correlations (pairwise potentials)

Example: Semantic segmentation





- A classifier provides evidence for every pixel (unary potentials)
- Smoothness is encouraged via correlations (pairwise potentials)

How much weight should we give to each of the two cues?

Feature vector for structured objects:

$f(x^{(i)}, \boldsymbol{y}) =$

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \left[\begin{array}{c} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{array}\right] =$$

Feature vector for structured objects:

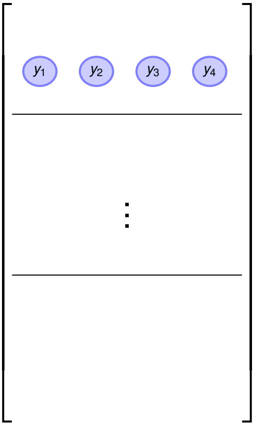
$$f(x^{(i)}, \boldsymbol{y}) = \left[\begin{array}{c} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{array}\right] = \left[\begin{array}{c} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{array}\right] =$$

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \left[\begin{array}{c} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{array}\right] = \left[\begin{array}{c} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{array}\right] = \left[\begin{array}{c} \\ \\ \\ \hline \vdots \\ \hline \\ \\ \\ \end{array}\right]$$

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \begin{bmatrix} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{bmatrix} =$$

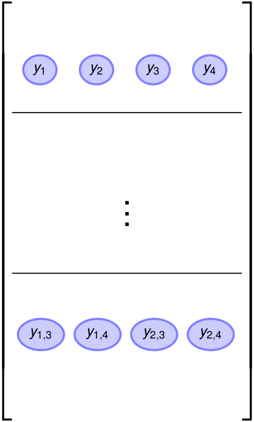


Feature vector for structured objects:

$$f(x^{(i)},\boldsymbol{y}) = \left[\begin{array}{c} f_1(x^{(i)},\boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)},\boldsymbol{y}) \end{array}\right] = \left[\begin{array}{c} \sum_r f_{1,r}(x^{(i)},\boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)},\boldsymbol{y}_r) \end{array}\right] = \left[\begin{array}{cccc} y_1 & y_2 & y_3 & y_4 \\ \hline & \vdots & & \\ \hline y_{1,3} & y_{1,4} & y_{2,3} & y_{2,4} \end{array}\right]$$

Feature vector for structured objects:

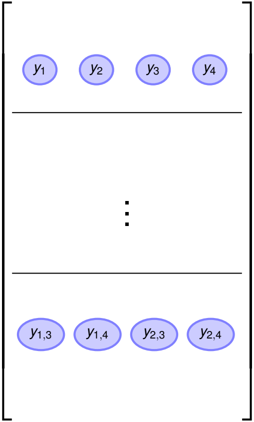
$$f(x^{(i)}, \mathbf{y}) = \begin{bmatrix} f_1(x^{(i)}, \mathbf{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \mathbf{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \mathbf{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \mathbf{y}_r) \end{bmatrix} =$$



- Every vector element f_m is a graphical model

Feature vector for structured objects:

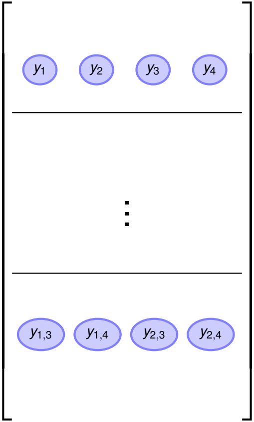
$$f(x^{(i)}, \mathbf{y}) = \begin{bmatrix} f_1(x^{(i)}, \mathbf{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \mathbf{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \mathbf{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \mathbf{y}_r) \end{bmatrix} =$$



- Every vector element f_m is a graphical model
- Learn to weight parts of the combined graphical model

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \begin{bmatrix} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{bmatrix} =$$

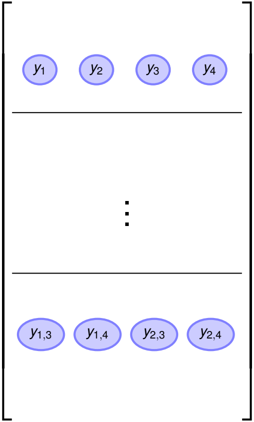


- Every vector element f_m is a graphical model
- Learn to weight parts of the combined graphical model

$$F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}) = \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}) =$$

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \begin{bmatrix} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{bmatrix} =$$



- Every vector element f_m is a graphical model
- Learn to weight parts of the combined graphical model

$$F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}) = \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}) = \sum_{m=1}^{M} w_m \underbrace{\left(\sum_r f_{m,r}(x^{(i)}, \boldsymbol{y}_r) \right)}_{\text{individual models}} =$$

Feature vector for structured objects:

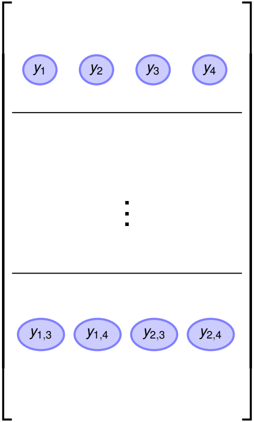
$$f(x^{(i)}, \boldsymbol{y}) = \begin{bmatrix} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{bmatrix} = \begin{bmatrix} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{bmatrix} = \begin{bmatrix} y_1 \quad y_2 \quad y_3 \quad y_4 \\ \hline \vdots \\ \hline y_{1,3} \quad y_{1,4} \quad y_{2,3} \quad y_{2,4} \end{bmatrix}$$

- Every vector element f_m is a graphical model
- Learn to weight parts of the combined graphical model

$$F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}) = \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}) = \sum_{m=1}^{M} w_m \underbrace{\left(\sum_r f_{m,r}(x^{(i)}, \boldsymbol{y}_r) \right)}_{\text{individual models}} = \sum_r \hat{f}_r(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}_r)$$

Feature vector for structured objects:

$$f(x^{(i)}, \boldsymbol{y}) = \left[\begin{array}{c} f_1(x^{(i)}, \boldsymbol{y}) \\ \hline \vdots \\ \hline f_M(x^{(i)}, \boldsymbol{y}) \end{array}\right] = \left[\begin{array}{c} \sum_r f_{1,r}(x^{(i)}, \boldsymbol{y}_r) \\ \hline \vdots \\ \hline \sum_r f_{M,r}(x^{(i)}, \boldsymbol{y}_r) \end{array}\right] =$$



- Every vector element f_m is a graphical model
- Learn to weight parts of the combined graphical model

$$F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}) = \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}) = \sum_{m=1}^{M} w_m \underbrace{\left(\sum_r f_{m,r}(x^{(i)}, \boldsymbol{y}_r)\right)}_{\text{individual models}} = \underbrace{\sum_r \hat{f}_r(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}_r)}_{\text{combined model}}$$

Recall: How did we derive multi-class SVM objective?

Recall: How did we derive multi-class SVM objective?

Learning target with groundtruth $\boldsymbol{y}^{(i)}$:

Recall: How did we derive multi-class SVM objective?

Learning target with groundtruth $\boldsymbol{y}^{(i)}$:

$$\forall \hat{\boldsymbol{y}} \quad \boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) \leq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

Recall: How did we derive multi-class SVM objective?

Learning target with groundtruth $\boldsymbol{y}^{(i)}$:

$$\forall \hat{\boldsymbol{y}} \quad \boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) \leq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

How many constraints are there?

Recall: How did we derive multi-class SVM objective?

Learning target with groundtruth $\boldsymbol{y}^{(i)}$:

$$\forall \hat{\boldsymbol{y}} \quad \boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) \leq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

How many constraints are there?

Alternatively:

$$\max_{\hat{\boldsymbol{y}}} \boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) \leq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

Hinge loss:

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})\right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

- Max-Margin Markov Network [Taskar et al. 2003]

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

- Max-Margin Markov Network [Taskar et al. 2003]
- Structured Support Vector Machine [Tsochantaridis et al. 2004]

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

- Max-Margin Markov Network [Taskar et al. 2003]
- Structured Support Vector Machine [Tsochantaridis et al. 2004]

$$\min_{\boldsymbol{w}} \frac{C}{2} \|\boldsymbol{w}\|_2^2 + \sum_{i \in \mathcal{D}} \left(\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) - \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)}) \right)$$

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

- Max-Margin Markov Network [Taskar et al. 2003]
- Structured Support Vector Machine [Tsochantaridis et al. 2004]

$$\min_{\boldsymbol{w}} \frac{C}{2} \|\boldsymbol{w}\|_2^2 + \sum_{i \in \mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)} - \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)}) \right)$$

Hinge loss:

Linearly penalize whenever maximum is within a margin $L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)})$ of the data $(x^{(i)}, \boldsymbol{y}^{(i)})$ score:

$$\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right) \geq \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)})$$

- Max-Margin Markov Network [Taskar et al. 2003]
- Structured Support Vector Machine [Tsochantaridis et al. 2004]

$$\min_{\boldsymbol{w}} \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i \in \mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x^{(i)}, \boldsymbol{y}^{(i)}) \right)$$

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

Iterate:

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

Iterate:

1. Loss-augmented inference:

$$\arg\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)$$

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

Iterate:

1. Loss-augmented inference:

$$\arg\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)$$

2. Perform gradient step:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla_{\boldsymbol{w}} L(\boldsymbol{w})$$

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i\in\mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

Iterate:

1. Loss-augmented inference:

$$\arg\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)$$

2. Perform gradient step:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla_{\boldsymbol{w}} L(\boldsymbol{w})$$

How complicated is this?

How to optimize this?

$$\min_{\boldsymbol{w}} L(\boldsymbol{w}) := \frac{C}{2}\|\boldsymbol{w}\|_2^2 + \sum_{i \in \mathcal{D}} \left(\underbrace{\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)}_{\text{Loss-augmented inference}} - \boldsymbol{w}^\top f(x, \boldsymbol{y}^{(i)}) \right)$$

E.g., with (sub-)gradient descent:

Iterate:

1. Loss-augmented inference:

$$\arg\max_{\hat{\boldsymbol{y}}} \left(\boldsymbol{w}^\top f(x^{(i)}, \hat{\boldsymbol{y}}) + L(\hat{\boldsymbol{y}}, \boldsymbol{y}^{(i)}) \right)$$

2. Perform gradient step:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla_{\boldsymbol{w}} L(\boldsymbol{w})$$

How complicated is this?

Solve one structured prediction task per sample per iteration.

Structured Learning

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

Structured Learning

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

Subsumes:

Structured Learning

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

Subsumes:

- linear structured SVMs

Structured Learning

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

Subsumes:

- linear structured SVMs
- linear conditional random fields (we condition on data x)

Structured Learning

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

Subsumes:

- linear structured SVMs
- linear conditional random fields (we condition on data x)
- binary/multi-class SVM

Structured Learning

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

Subsumes:

- linear structured SVMs
- linear conditional random fields (we condition on data x)
- binary/multi-class SVM
- logistic/multi-class regression

Structured Learning

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

Subsumes:

- linear structured SVMs
- linear conditional random fields (we condition on data x)
- binary/multi-class SVM
- logistic/multi-class regression

What changed compared to our previous formulation?

Structured Learning

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

What's the main limitation?

What happens if we make our score function non-linear in its parameters $\boldsymbol{w}$?

What happens if we make our score function non-linear in its parameters $\boldsymbol{w}$?

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

What happens if we make our score function non-linear in its parameters $\boldsymbol{w}$?

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

What happens if we make our score function non-linear in its parameters $\boldsymbol{w}$?

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

$F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$ is a score function represented by a computation graph, e.g., a deep net

How to optimize this program:

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

How to optimize this program:

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

How to optimize this program:

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

Let's try gradient descent: (for simplicity $\epsilon = 1$)

How to optimize this program:

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

Let's try gradient descent: (for simplicity $\epsilon = 1$)

$$p(\hat{\boldsymbol{y}}; x, \boldsymbol{w}) = \frac{\exp F(\boldsymbol{w}, x, \hat{\boldsymbol{y}})}{\sum_{\tilde{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x, \tilde{\boldsymbol{y}})}$$

How to optimize this program:

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

Let's try gradient descent: (for simplicity $\epsilon = 1$)

$$p(\hat{\boldsymbol{y}}; x, \boldsymbol{w}) = \frac{\exp F(\boldsymbol{w}, x, \hat{\boldsymbol{y}})}{\sum_{\tilde{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x, \tilde{\boldsymbol{y}})}$$

$$\begin{aligned} & C\boldsymbol{w} + \sum_{i\in\mathcal{D}} \left(\sum_{\hat{\boldsymbol{y}}} p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right) \\ = \; & C\boldsymbol{w} + \sum_{i\in\mathcal{D}, \hat{\boldsymbol{y}}} \left(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}) \right) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) \end{aligned}$$

How to optimize this program:

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

Let's try gradient descent: (for simplicity $\epsilon = 1$)

$$p(\hat{\boldsymbol{y}}; x, \boldsymbol{w}) = \frac{\exp F(\boldsymbol{w}, x, \hat{\boldsymbol{y}})}{\sum_{\tilde{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x, \tilde{\boldsymbol{y}})}$$

$$\begin{aligned} & C\boldsymbol{w} + \sum_{i\in\mathcal{D}} \left(\sum_{\hat{\boldsymbol{y}}} p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right) \\ = \; & C\boldsymbol{w} + \sum_{i\in\mathcal{D},\hat{\boldsymbol{y}}} \Big(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \underbrace{\delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)})}_{p^{\text{GT}}_{\boldsymbol{y}^{(i)}}(\hat{\boldsymbol{y}})} \Big) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) \end{aligned}$$

Algorithm

Repeat until stopping criteria

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$
2. Compute $p(\hat{\boldsymbol{y}}; x, \boldsymbol{w})$ via soft-max

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$
2. Compute $p(\hat{\boldsymbol{y}}; x, \boldsymbol{w})$ via soft-max
3. Backward pass via chain rule to obtain gradient

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$
2. Compute $p(\hat{\boldsymbol{y}}; x, \boldsymbol{w})$ via soft-max
3. Backward pass via chain rule to obtain gradient
4. Update parameters $\boldsymbol{w}$

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$
2. Compute $p(\hat{\boldsymbol{y}}; x, \boldsymbol{w})$ via soft-max
3. Backward pass via chain rule to obtain gradient
4. Update parameters $\boldsymbol{w}$

Where are the challenges?

Algorithm

Repeat until stopping criteria

1. Forward pass to compute $F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$
2. Compute $p(\hat{\boldsymbol{y}}; x, \boldsymbol{w})$ via soft-max
3. Backward pass via chain rule to obtain gradient
4. Update parameters $\boldsymbol{w}$

Where are the challenges?

- How do we represent $F(\boldsymbol{w}, x, \hat{\boldsymbol{y}})$ if output space $\mathcal{Y}$ is large?

Output space size of some applications:

Output space size of some applications:

Tag prediction

Output space size of some applications:

Tag prediction

$$|\mathcal{Y}| = 2^{\#tags}$$

Output space size of some applications:

Tag prediction Segmentation

$|\mathcal{Y}| = 2^{\#tags}$

Output space size of some applications:

Tag prediction | Segmentation

$|\mathcal{Y}| = 2^{\#tags}$ $\quad$ $|\mathcal{Y}| = C^{\#pixels}$

Output space size of some applications:

Tag prediction	Segmentation
	
$\lvert\mathcal{Y}\rvert = 2^{\#tags}$	$\lvert\mathcal{Y}\rvert = C^{\#pixels}$

Observation:

- Interest in jointly predicting multiple variables $\boldsymbol{y} = (y_1, \ldots, y_D)$

Output space size of some applications:

Tag prediction | Segmentation

$|\mathcal{Y}| = 2^{\#tags}$ $\quad$ $|\mathcal{Y}| = C^{\#pixels}$

Observation:

- Interest in jointly predicting multiple variables $\boldsymbol{y} = (y_1, \ldots, y_D)$

Solution:

- assume scoring function decomposes additively

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

Output space size of some applications:

Tag prediction | Segmentation

$|\mathcal{Y}| = 2^{\#tags}$ $\qquad$ $|\mathcal{Y}| = C^{\#pixels}$

Observation:

- Interest in jointly predicting multiple variables $\boldsymbol{y} = (y_1, \ldots, y_D)$

Solution:

- assume scoring function decomposes additively

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

Every $f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$ is an arbitrary composite function, e.g., a CNN

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

How to compute gradient:

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{w}} \quad & \sum_{i \in \mathcal{D}} \left(\log \sum_{\hat{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right) \\
& = \sum_{i \in \mathcal{D}, \hat{\boldsymbol{y}}} \left(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}) \right) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})
\end{aligned}$$

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

How to compute gradient:

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{w}} \quad & \sum_{i \in \mathcal{D}} \left(\log \sum_{\hat{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right) \\
& = \sum_{i \in \mathcal{D}, \hat{\boldsymbol{y}}} \left(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}) \right) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) \\
& = \sum_{i \in \mathcal{D}, r, \hat{\boldsymbol{y}}_r} \left(p_r(\hat{\boldsymbol{y}}_r; x^{(i)}, \boldsymbol{w}) - \delta_r(\hat{\boldsymbol{y}}_r = \boldsymbol{y}_r{}^{(i)}) \right) \nabla_{\boldsymbol{w}} f_r(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}_r)
\end{aligned}$$

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

How to compute gradient:

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{w}} \quad & \sum_{i \in \mathcal{D}} \left(\log \sum_{\hat{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right) \\
& = \sum_{i \in \mathcal{D}, \hat{\boldsymbol{y}}} \left(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}) \right) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) \\
& = \sum_{i \in \mathcal{D}, r, \hat{\boldsymbol{y}}_r} \left(p_r(\hat{\boldsymbol{y}}_r; x^{(i)}, \boldsymbol{w}) - \delta_r(\hat{\boldsymbol{y}}_r = \boldsymbol{y}_r{}^{(i)}) \right) \nabla_{\boldsymbol{w}} f_r(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}_r)
\end{aligned}$$

How to obtain marginals $p_r(\hat{\boldsymbol{y}}_r; x, \boldsymbol{w})$?

$$F(\boldsymbol{w}, x, \boldsymbol{y}) = F(\boldsymbol{w}, x, y_1, \ldots, y_D) = \sum_r f_r(\boldsymbol{w}, x, \boldsymbol{y}_r)$$

How to compute gradient:

$$\frac{\partial}{\partial \boldsymbol{w}} \quad \sum_{i \in \mathcal{D}} \left(\log \sum_{\hat{\boldsymbol{y}}} \exp F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}) - F(\boldsymbol{w}, x^{(i)}, \boldsymbol{y}^{(i)}) \right)$$

$$= \sum_{i \in \mathcal{D}, \hat{\boldsymbol{y}}} \left(p(\hat{\boldsymbol{y}}; x^{(i)}, \boldsymbol{w}) - \delta(\hat{\boldsymbol{y}} = \boldsymbol{y}^{(i)}) \right) \nabla_{\boldsymbol{w}} F(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}})$$

$$= \sum_{i \in \mathcal{D}, r, \hat{\boldsymbol{y}}_r} \left(p_r(\hat{\boldsymbol{y}}_r; x^{(i)}, \boldsymbol{w}) - \delta_r(\hat{\boldsymbol{y}}_r = \boldsymbol{y}_r^{(i)}) \right) \nabla_{\boldsymbol{w}} f_r(\boldsymbol{w}, x^{(i)}, \hat{\boldsymbol{y}}_r)$$

How to obtain marginals $p_r(\hat{\boldsymbol{y}}_r; x, \boldsymbol{w})$?

Approximate marginals $b_r(\hat{\boldsymbol{y}}_r; x, \boldsymbol{w})$ via:

- LP relaxation
- Message passing

General algorithm for learning:

repeat

  repeat

    update marginals b_r

  until convergence

  update parameters $\boldsymbol{w}$

until convergence

General algorithm for learning:

repeat

    repeat

        update marginals b_r

    until convergence

    update parameters $\boldsymbol{w}$

until convergence

What operation is used for computation of marginals in multi-class prediction?

General algorithm for learning:

repeat

  repeat

    update marginals b_r

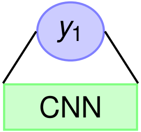
  until convergence

  update parameters $\boldsymbol{w}$

until convergence

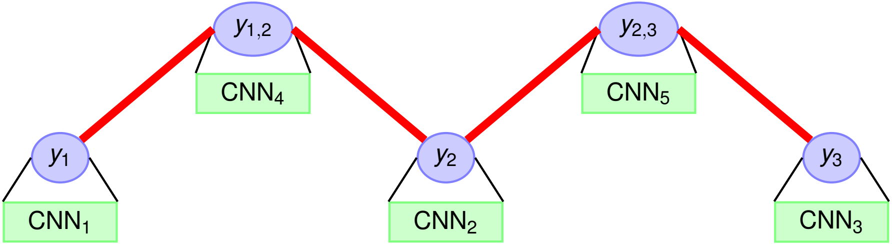
What operation is used for computation of marginals in multi-class prediction? Soft-max

Intuition: Standard CNN



Intuition: Independent Prediction

Intuition: Deep Structured Learning



Algorithm summary:

Repeat until stopping criteria

Algorithm summary:

Repeat until stopping criteria

1. Forward passes to compute $f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$

Algorithm summary:

Repeat until stopping criteria

1. Forward passes to compute $f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$
2. **Estimate beliefs** $b_r(\hat{\boldsymbol{y}}_r, x, \boldsymbol{w})$ **exactly/approximately**

Algorithm summary:

Repeat until stopping criteria

1. Forward passes to compute $f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$
2. **Estimate beliefs** $b_r(\hat{\boldsymbol{y}}_r, x, \boldsymbol{w})$ **exactly/approximately**
3. Compute difference between estimated and groundtruth beliefs

Algorithm summary:

Repeat until stopping criteria

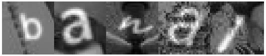
1. Forward passes to compute $f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$
2. **Estimate beliefs** $b_r(\hat{\boldsymbol{y}}_r, x, \boldsymbol{w})$ **exactly/approximately**
3. Compute difference between estimated and groundtruth beliefs
4. Backpropagation of difference to obtain gradient

Algorithm summary:

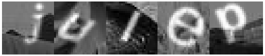
Repeat until stopping criteria

1. Forward passes to compute $f_r(\boldsymbol{w}, x, \hat{\boldsymbol{y}}_r)$
2. **Estimate beliefs** $b_r(\hat{\boldsymbol{y}}_r, x, \boldsymbol{w})$ **exactly/approximately**
3. Compute difference between estimated and groundtruth beliefs
4. Backpropagation of difference to obtain gradient
5. Update parameters $\boldsymbol{w}$

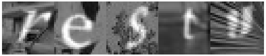
Example: Find five letters within distorted images



banal

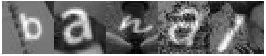


julep

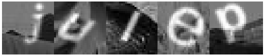


resty

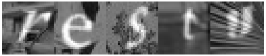
Example: Find five letters within distorted images



banal

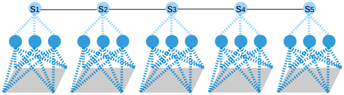


julep

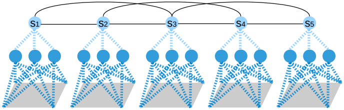


resty

- Graphical model: 1st or 2nd order Markov

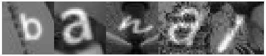


1st order Markov

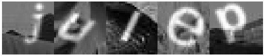


2nd order Markov

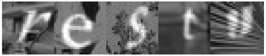
Example: Find five letters within distorted images



banal

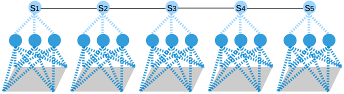


julep

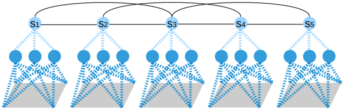


resty

- Graphical model: 1st or 2nd order Markov

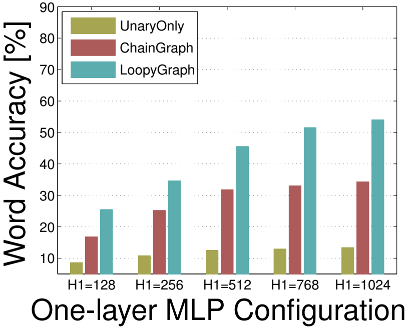


1st order Markov

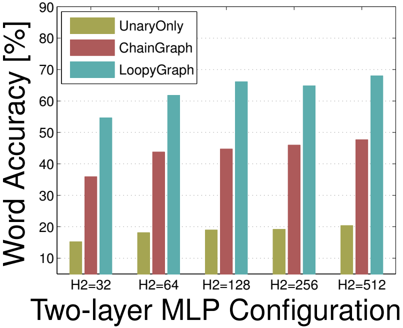


2nd order Markov

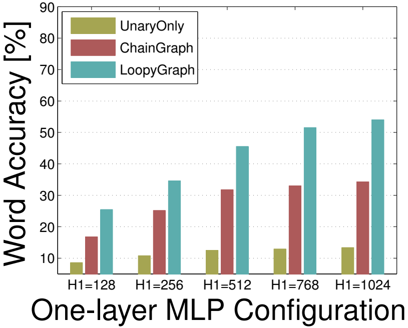
- Unary function: multi-layer perceptron (MLP)
- Pairwise function: linear or non-linear MLP
- $|\mathcal{Y}| = 26^5$

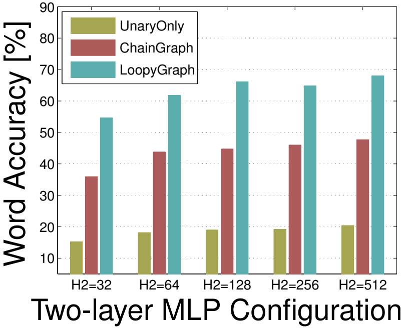


Word Accuracy [%]
UnaryOnly
ChainGraph
LoopyGraph
H1=128
H1=256
H1=512
H1=768
H1=1024
One-layer MLP Configuration



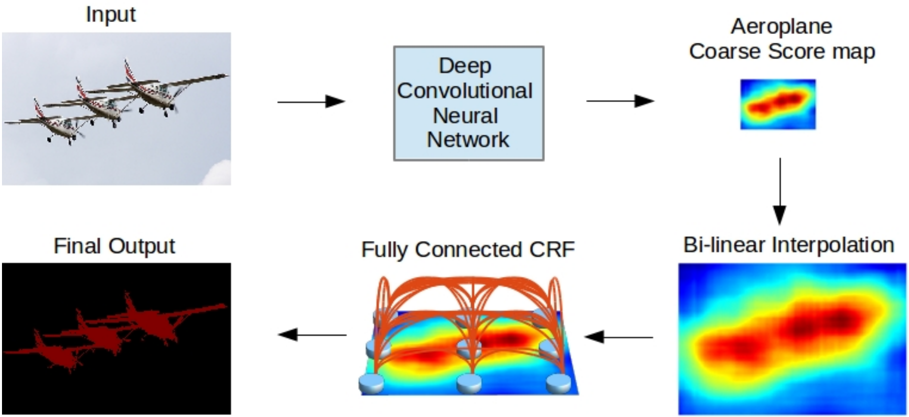
Word Accuracy [%]
UnaryOnly
ChainGraph
LoopyGraph
H2=32
H2=64
H2=128
H2=256
H2=512
Two-layer MLP Configuration





Deeper and more structured $\Rightarrow$ better performance

Example: Semantic segmentation with DeepLab



Results:


(a) Image
(b) Before CRF
(c) After CRF
(a) Image
(b) Before CRF
(c) After CRF
(a) Image
(b) Before CRF
(c) After CRF

Recap:

Recap:

Inference:

Recap:

Inference:

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

Recap:

Inference:

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

Learning:

Recap:

Inference:

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

Learning:

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

Recap:

Inference:

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

Learning:

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

repeat
- repeat
  - update marginals b_r
- until convergence
- update parameters $\boldsymbol{w}$

until convergence

Quiz:

Quiz:

- What's our feature vector for structured output space data?

Quiz:

- What's our feature vector for structured output space data?
- Describe algorithms for learning with structured output space data?

Quiz:

- What's our feature vector for structured output space data?
- Describe algorithms for learning with structured output space data?
- What makes learning with structured output spaces hard?

Quiz:

- What's our feature vector for structured output space data?
- Describe algorithms for learning with structured output space data?
- What makes learning with structured output spaces hard?
- How can deep nets and structured output spaces be combined?

Important topics of this lecture

- Feature vectors for structured output space data
- Learning with structured output space data

Up next:

- Learning Theory